Particle

Animation

Software for

LightWave 3D™

SPARKS

users manual

METRO graf
Animation and Software

SPARKS

users manual

METRO graf
Animation and Software

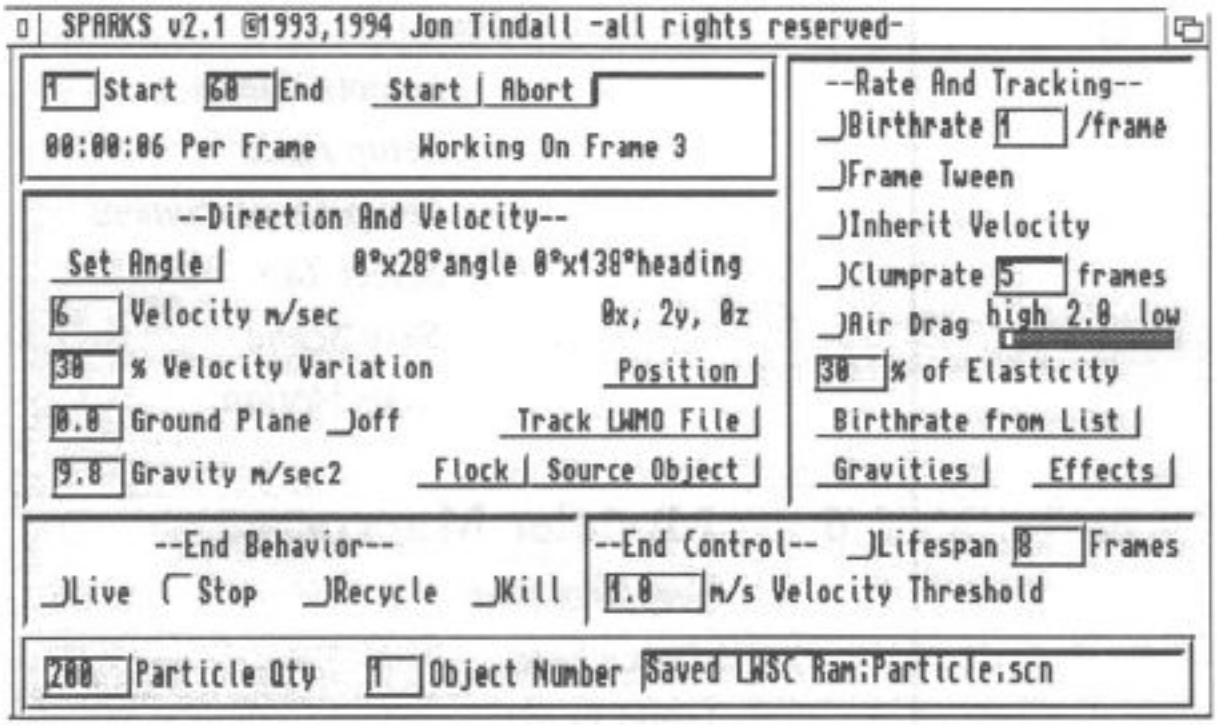
Copyright

Published by MetroGrafx
Printed in USA

MetroGrafx
625 Newton Dr.
Lake Orion, Mi. 48362
(810) 693-5134
Tech support from mon. thru fri. 10:00am to 4:00pm est.

Special thanks to [my backers],
Cheryl Tindall
Ray Tindall
Annette and Eric Tindall
Hank and Jennette Groth
Dale Myers
Paul Tyson
Tom Richardson
Jim Arthurs
And all the buyers and supporters of SPARKS,
may particles fly forth from your fingertips.

Contents



The **SPARKS** program main interface

1 Introduction

Congratulations on your purchase of SPARKS.
SPARKS represents the cutting edge of procedural animation. Effects previously proprietary and available only on high-end animation software is now within your reach. Combined with LightWave 3D you have access to unbelievably complex and beautiful movement of both particles and models.

We have gone to great lengths to assure that this power is easy and fun to use for both the novice and the professional animator.

SYSTEM REQUIREMENTS:

Amiga 2000, 3000 or 4000.
Workbench 2.1 or higher.
Lightwave 3D, v3.0 or higher.
Recommended:
an 030 or 040 Accelerator
16MB of Fast Ram
A Large HardDrive

Whats New for version 2.10?

The object requester in the scene menu has new functionality:

By Shift-selecting *del* you can delete the object list entirely.
By Shift-Selecting *add* you can select an entire directory of models at once. Perfect for all those pieces that fragment.lwm produces.

The Save and Load list save routine now saves rotation evaluations as well.

A new random rotation feature in the *Select Object* requester is enabled by entering "r" into the *evalH, evalP, or evalB* fields in the *Select Object* requester. Objects will have a random rotation applied to them automatically. Perfect for explosions. The default maximum rotation per frame is 10. To change it just enter a number after entering the r, i.e. "r 20" this would set the maximum rotation per frame to 20. So an object may be at best rotate 360° (over 18 frames). You can make it different for each axis as well.

Gravity requester is now a database style interface.

Gravity requester now accommodates different fonts in the title bar.

The *Source Object* requester has had a face lift. Options meaningless to the current selection are not shown. Sort option is gone, it is not needed anymore (Modeler 3.1's ARexx point ordering has been fixed).

The *angle requester* has been changed so that the nozzle can no longer point "backwards" as this has been a source of confusion.

The big one I know you've all been waiting for....Spline interpolation! The *Pathfreezer.lwm* was all fine and good EXCEPT when I went to add the ability to point the "nozzle" or origin according to a LWMO file setup in LightWave and saved via save motion. That's when I noticed there was no way to get the rotations of the object back out of the Modeler and into the LWMO file that was being built. Dang! Back to the books for a couple of weeks to learn LightWave's particular flavor of spline that objects move along. Now anytime a Motion file is input into **SPARKS** it is interpolated. Of course now I have the rotations I need as well as the scale. So not only will the origin move, but the "nozzle" can turn (you will need to restrict the spread nozzle through the angle control in order to be viewable) and the velocity can be altered by scaling the object. The nozzle points toward plus x axis normally. The velocity is a

straight linear scale (if object is 0.5 the velocity is divided by 2).
A new targeting option is presented after choosing either *Source Object* or *Track LWMO file* that allows selecting a motion file as a target. This allows for moving targets. The target represents an initial directional vector only. In other words you might not even come close to the target unless the velocity was high enough to allow it. If you want to hit something you may have to lead it an appropriate amount.

A new item in the scene menu will parent all objects to a null at 0,0,0. This was a much requested feature that allows you to move all objects in the scene at one time as well as rescale.them, delete them and parent them to other things. The object is "hardwired" to be Objects:NullObject. If you do not have one there just load anything your heart desires when told that Objects:NullObject can't be found. Version 3.1 or better of LightWave will load a null automatically if it you hit cancel when it says it can't find an object.

A new AutoLoad toggle in the control menu allows for automatic loading into LightWave after a scene file is created.

5 New macros for the Modeler:

PathFreezer.lwm	...Makes a key at every frame
GetSource.lwm	...Fetch the source file from t:
GetTarget.lwm	...Fetch the target file from t:
Subdivided.lwm	...Mesh out an object using slice
Fragment.lwm	...Divide your model and save

1.1 Whats a Particle?

A particle is a dimensionless point. It takes up no space (has no volume) and can be placed anywhere in 3D space by giving an x,y,z coordinate. They are an ideal candidate for physics based calculations. In LightWave they can be constructed in the modeler by making a single vertices, selecting it and making a polygon. This single point polygon has special meaning to LightWave. LightWave has an advanced renderer that can take these single point polygons and render them as a single pixel no matter how close or far away they are from the camera. They are also very light in terms of memory usage and rendering time. They can be used by the thousands to provide effects that can not be done any other way. The starfield objects included with LightWave is a example of a particle object. The snowcube and raincube also included with LightWave are others.

Rotation angles make no sense for a particle. When you make a point in Modeler you do not need to enter an angle, only a x, y and z value. Through the miracle of modern science we have added the ability to create angles of rotation. They are built when you write a scene file according to a formula entered in the *Select Object* requester. Of course, rotations only make sense for a object replacement that takes up space not a single point polygon.

Particles cannot collide. See, I told you they were convenient. This saves tons of calculations, besides the fact that real collision detection brings the mightiest machines to their knees. But it it easy and fast to have a ground plane the particles can bounce off. The ground plane is also handy to as a way to stop a particle so it can be recycled.

1.2 What is Procedural Animation?

Since this program was designed to work in conjunction with LightWave, you have probably already had some experience with key frame animation. In key frame animation you set up a start and end position for your models as well as the time that the move is to take place and the computer automatically interpolates the in-between positions where there is no key frame data.

With procedural animation you simply setup the initial start condition and the computer takes over from there.An algorithm is applied that can predict where the particle will be in the next frame by where it has been on the last! In other words it knows the velocity and directional information of each particle in your scene. Using that information SPARKS is able to use ARexx to position the particle at each frame and tell LightWave to render it.

There are some tradeoffs in procedural animation though. For starters once the particle is let loose it is out of your direct control as it has a mind of its own. It also has to run like a program from start to finish because it is a linear computation that works out each frame in order. You can save your settings but don't expect any 2 sessions to produce exactly the same results because of the randomizational effects of the control set. The beauty of this is truly organic looking motion. Although you loose direct control, don't think you are left out of the loop. You still need to visualize the controls necessary to guide the particles through their journey. These controls consist of direction and velocity as well as the birthrate, origin, groundplane position, gravitational forces, wind, return of energy on impact and a host of others. These controls will enable you to force thousands of particles to obey your every whim.

But the scope of this software does not stop there, it can replace those particles with objects of your choice. This powerful addition can place hundreds of models at start positions automatically, then animate their movement and rotations *all without even making a manual key frame*!

2 Getting Started

It is helpful to understand the concept of how SPARKS works and deals with particles. SPARKS works entirely within its own memory and does not require LightWave to operate. However, you will usually want them running together. SPARKS can work in a couple ways.

You cannot see the point movement as a preview animation because SPARKS must internally calculates the movement and path of every particle in the animation at each frame. Imagine a explosion scene with hundreds, even thousands of particles all going there separate direction and each with their own motion path. It is difficult for any PC to generate that kind of numeric calculation in near real time regardless of the software you are running.

The LightWave screen however allows you to watch the calculations take place to give you an idea of what's going on. By using LightWaves ARexx port we can specify a object and move its vertices one by one at each frame. Then tell LightWave to update the display. This is the default. You can shut this update feature off and gain some extra speed by toggling the pulldown control menu *No Move*. You can also define objects to replace the particles used in the calculation. SPARKS will record the particles path and write out a scene file to be loaded into LightWave and previewed or rendered.

SPARKS can:

Manipulate the vertices of one object. If these vertices are single point polygons they will be rendered as a visible particle. One object is selected as the raw material to provide point fodder for the cannon. This point object is loaded in, its points are relocated according to the origin setting. Then according to the rest of the settings its position is updated at each frame. You cannot see this point movement as a preview animation because the movement is calculated at each frame, updated by Lightwave then rendered or just viewed in the layout. The object is actually being changed in shape so be careful not to save it and overwrite a file you need unintentionally.

Or:

As your particles travel thru time and space they can be recorded

and written to a scene file along with the load path to the models you would like to replace them with. This enables bulk loading of objects placing them automatically at the position of choice. This is a powerful feature allowing for flocking, swarms and arrays of objects. When you work this way you are limited by the amount of memory in your machine, after all you must load up multiple models and they have keys at every frame. However by working this way you can make an animation preview in wireframe for realtime playback. You can also move your camera and view or render the scene from every angle after the simulation has been calculated and saved. As well as add it to a already built scene using the load *Object From Scene* button in LightWave. Both methods have their own advantages and a clever user can exploit them both to the upmost.

Let us digress for moment and discuss physics. What? You did'nt pay attention in science class? Nether did I. It seemed kinda remote at the time. Oh well! Now, however a program like SPARKS runs on physics. It can only respond to the numbers put into it. The first habit you should cultivate it to think in terms of scale. In 3D there is no reason why you can't make a ball the size of a planet. If you were to shoot this ball with a velocity of 30 meters per second and were looking at the entire ball at one time it would'nt even seem to move! Although if you were the size of an ant, 30 meters per second would seem quite speedy. LightWave and the Modeler (being the incredible programs they are) have given us a real scale (meters) to base the physics on. Some 3D programs only use unit measure (yuk!). One person thinks that a grid is inches and another person thinks it is millimeters, causing much confusion. Anyone who has brought a model in from one of those systems knows what I mean. Keeping a firm mental grasp on the scale issue will result in much more fruitful animating, rather than countless retries just fiddling with the numbers.

2.1 Starting SPARKS

The SPARKS program can be dragged to the directory of choice on your HD. SPARKS communicates to LightWave thru ARexx, please insure ARexx is up and running before attempting to run SPARKS.

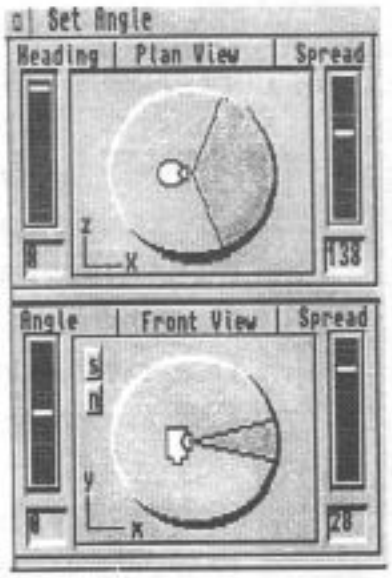
Start the Video Toaster software as you normally would. Its a good idea to launch the getsmall project upon entering the Video Toaster in the preferences requester. This will unload the effects banks and free up memory for your 3D work. SPARKS usually needs LightWave running so load it up and enter. There is a screen push shortcut you will find very useful for moving between LightWave's interface and SPARKS interface this is the left-Amiga m combination. That is hold down the left Amiga key and press m at the same time. This will push the current screen to the back allowing you to access the workbench. Start SPARKS. Now you are ready to animate!

3.1 Explosions

Lets get started with a simple explosion of particles. With every thing fired up, let's go to the Modeler and make some particles. We need some raw material not a finished object, so just use the *point distributions* macro to create 50 points. This macro takes care of making the single point polygons. The single point polygon can be used as a visual representation or they can be rendered at each frame. This can be verified after it is run by switching to poly select mode and selecting some. Select the *export* popup and highlight new. Just save to ram:points.lw for now.

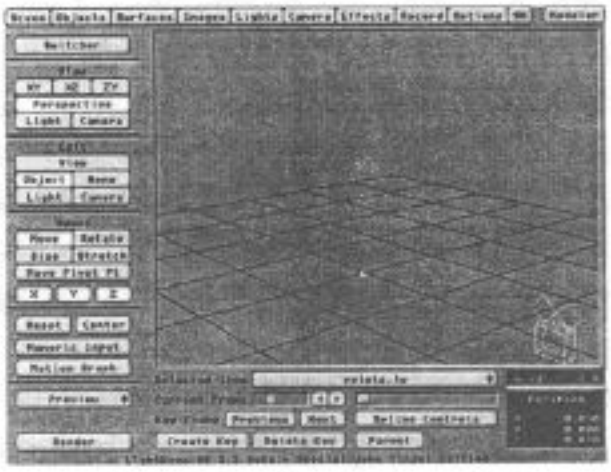
Go to LightWave. You should be able to see the particle cloud. Select *camera* edit and move the camera back to about Z=-12.0. Select rotate and rotate to around P=-7.0. Left-Amiga m back to the **SPARKS** interface.

The first thing to enter is the *point quantity*, set that to 50. The *object number* field is also important. If you had existing objects loaded into the scene this field would need to be change to its corresponding number in LightWave. If the point cloud was the fifth object loaded in LightWave the *object number* field would be 5. But it's object one now so leave it. Click on *Set Angle*, this brings up a requester to input the width and direction of the "nozzle" so to speak. The nozzle's default value is a spherical emission. It is read by looking at the front view window, the hilite is the angle and spread of the nozzle. Then the plan view window shows in hilite the direction and heading that the front view is swept through. The default settings are okay for now, close the window. There are several methods of controlling the start position. The quickest way is to enter it directly into the *Start Position* field. change the Y field to 2.5. This will set the center of the explosion to 2.5 meters above the 0. The rest of the settings can be left default. At This point you still have a few options to take care of. Chief among them is how to view this thing. Of course you can hit *Start*, let it render and just observe.



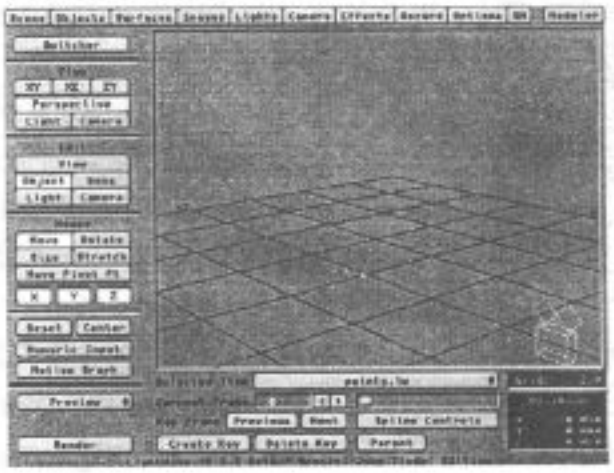
Or:

Option 1 is press no render. This will prevent the rendering step and is designed to calculate and and move and objects vertices for feedback, as well as way to bypass LightWaves limit on objects (version 1.0 = 100 objects,version 2.0 = 400 objects and version 3.0 = 1000 objects).



Option 2 is to select save rgb in LightWave and run the simulation saving the files to disk then compiling an animation or recording to tape.

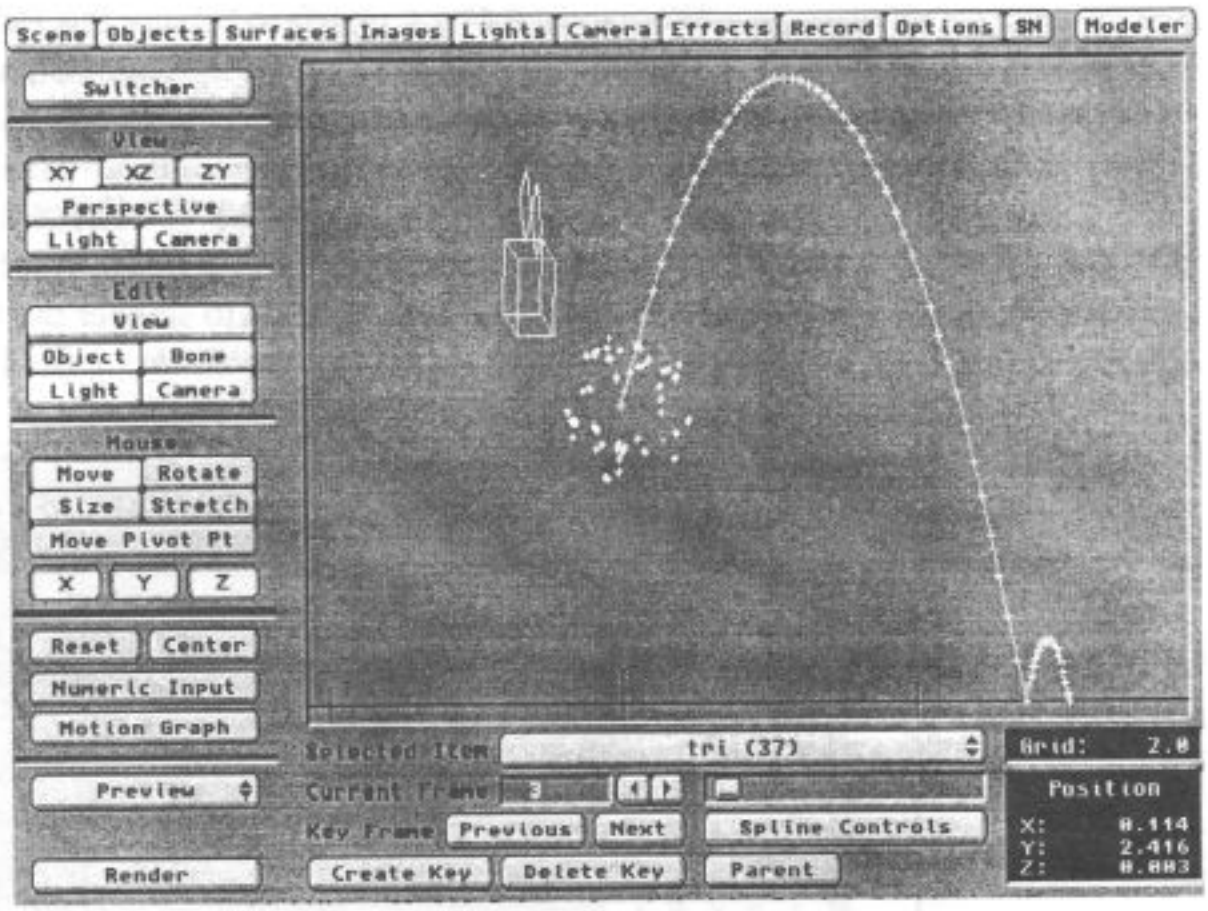
Option 3 is to select a object(s) to replace the particles with. This allows you to load the scene into LightWave and compile a preview animation and see it from any angle after the scene is created. Select *Select Objec*t from the scene pulldown menu. Click on *add* and select the particle.lw object from the SPARKS disk2 objects drawer. Select *Save scene* from menu and name the scene. The button *no render* is automatically selected. Selecting *no move* will speed things up. Change the value in the end frame field to 60. Clicking on *Start* will launch the process. Once the simulation is run. A scene file will be written to your choice of directories. Load this and you can make a preview as well as view it from different angles. Congratulations on making your first animation with SPARKS.



Try redoing this tutorial and changing some of the parameters to get a feel for what each one does.

Another thing to try is a multiple explosion. This can be achieved easily by going to LightWave. Load a null object, position it and make keys where you would like the explosions to occur. Click on

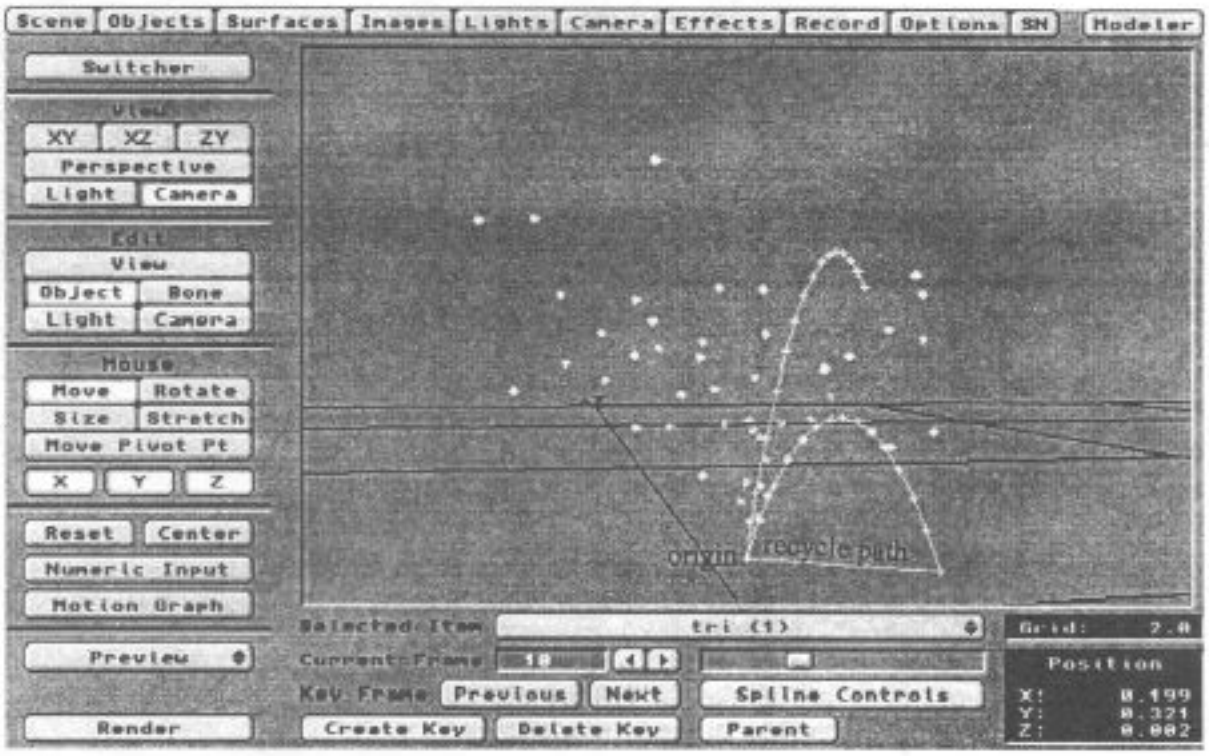
motion graph and save the motion file. Hot-key to **SPARKS**. Select *Birthrate from list* and an editor will come up allowing you to input the frame number and quantity of particles to released at that frame. Click on *Track LWMO* button and indicate the motion file you have just saved. Save scene file and click start. When finished, load the scene file into LightWave. Make a preview and play. You will see the particles are being released at the origin indicated by the motion file with the quantity entered in the birthrate editor.



3.2 Fountains

Let's try a fountain this time. 50 particles may not be enough for a smooth continuously flowing fountain but it should be enough for our purposes. You should be starting out with the one 50 point object we created from the last tutorial loaded in LightWave. If not load one up so we have some points to work with.

From the **SPARKS** interface be sure the *Point Quantity* field is set to 50 (the quantity of points to be moved) . Click on *position* to bring up the *Position* requester. Set the Y value to 0.1. This is of course .1 meters from the ground. We won't shoot these particles too high so scale back the default velocity by setting the *Velocity* field to 3 meters per second (m/s). The default setting for *Birthrate* is off. This is fine for hard explosions but for fountains we need to stagger the launch time and produce a stream of particles and the *Birthrate* field lets us do just that.



Let's see, with 50 particles to work with, too high of a number will make a rather spurting fountain as all the available material would be used up in few frames. The *Birthrate* regulates the number of particles released per frame. So enter 3 for the *Birthrate*. When the particles hit the ground their velocity is checked against the *Velocity Theshold* value,

if it is lower they become inactive and they can be recycled back into the queue. Toggle on the *Recycle* button in the *End Behavior* area. By the time we hit frame 17 and all our particles are used up, some particles are running out of steam and being recycled back into the fountain.

The default is for all particles to bounce on the *Ground Plane* until their velocity falls below the value in the *Velocity Threshold* field. The *Velocity Threshold* can be adjusted to provide harder or softer stops to your particles. This number in m/s influence's a particles *end behavior* as it is what tells the software when a particular particle has reached it's *end state*. The *Velocity Threshold* can be left alone for now. Lets change the fact that all the particles bounce. By clicking on the *Effects* button a requester will come up displaying among other things a *Bounce Probability* field. Setting this to 10 will allow only a 10 percent probability of a particle bouncing.

The *Velocity Variation* field allows us to randomize the velocity over and under the base *Velocity* field. Lets set this number to 20. 20 percent of 3(meters per second) is .6m/s. So our particles will have a initial velocity of between 2.4m/s and 3.6m/s.

The only thing left is to point the fountain. Click on the *Set Angle* button the display the angle selection gadgets. Our fountain will have a narrow nozzle so set the *Angle spread* to 30° and angle to 75°. If you can imagine the highlited area in the front view swept through the heading highlighted in the plan view you can visualize the direction of the flow.

Set the *End* frame for the amount of frames to render. Now just calculate it out with *No Render* selected for about 20 frames or so and click on abort. Hot-key back to LightWave and position the camera for a good view. Go to the camera menu and select lo-res. Go to the record menu and setup a save rgb path. Hot-key back to **SPARKS** and turn off the *no render* option. Press *Start*.

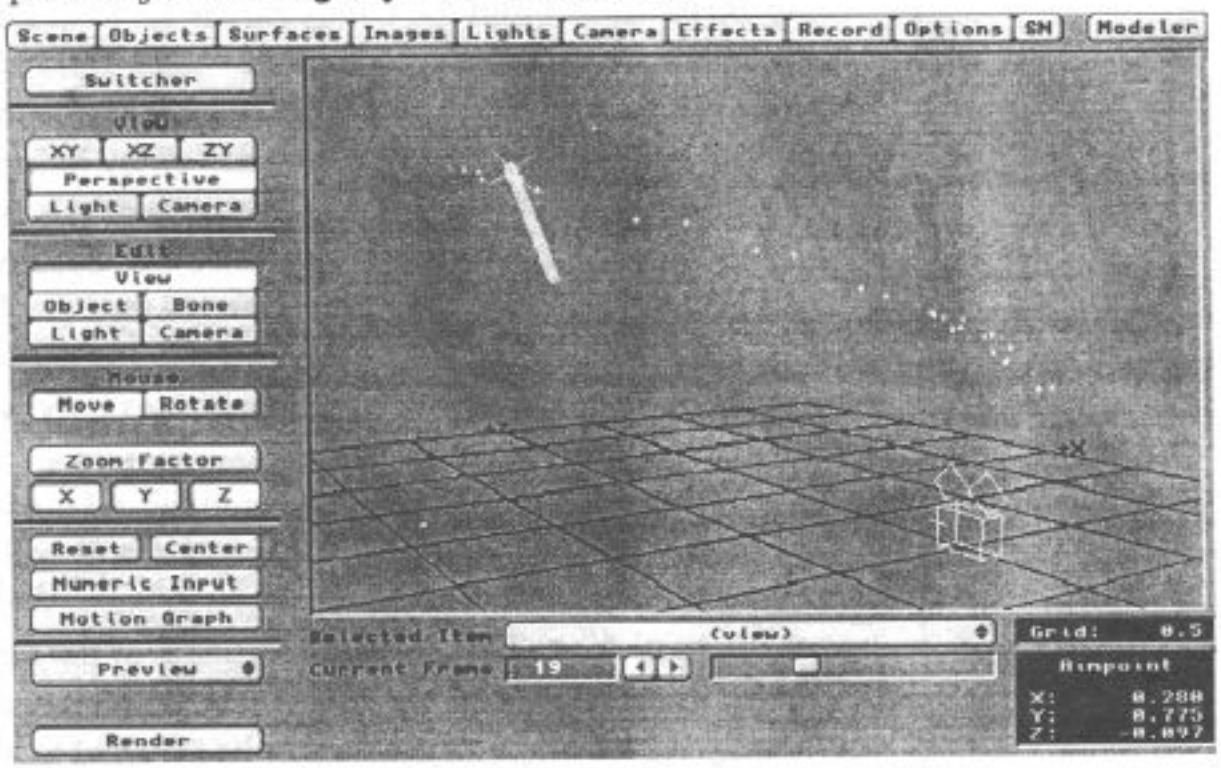
Particles render quick. After they are done, use ADPro or a similar utility to make a animation of your finished fountain. (NOTE: Amiga 4000 users can make a anim for playback from the switcher).

We have used the default surface for the last two tutorials. If you select and rename portions of your particle models polygons in the Modeler's surface requester then bring that in it will have different surfaces for you to apply colors to. All the mapping and surface values apply to particles and when the maps are locked to the world coordinates (see your LightWave manual for a description of this function) the particles will pass thru them changing in color as the map does.

3.3 Wand & Trail Tutorial

If your latest animation calls for a magic wand or fairy dust. Maybe even a trail of sparks from a rocket blasting an alien to dust, be sure to work thru this tutorial.

Lets begin by creating the wand in Modeler. Make a cylinder X=15mm, Y=600mm, Z=15mm, 2 segments, Bottom=0, Top=600. Select the points in the middle of the wand and move them up to the end to form a tip. Name the surfaces, wand and wandtip. Now go to Objects and select export new, save the wand to ram:wand. Go to Lightwave and set-up a motion file for the wand to wave, you know like a magician waves his magic wand. Now you must produce a motion file for your moving point of origin. That is, load a null object or such and position it at the tip of the wand and make a key at every frame until you have released all your particles, tracking the tip of the wand all the way. This produces a motion file containing the path of the particles release origin for us to use in our calculations. Save it for use in **SPARKS**. This step is a royal pain, but can be done in a reasonable amount of time providing that you dont need hundreds of keys. The trick is tracking the wand and setting up keys in one view then going to another view and repeating the process just making keys for the last axis.



Another easier approch is to simply set the wands local 0,0,0 to be at the wands tip in the Modeler. This way the origin would be coincident with the wands motion file. This method puts the pivot point at the wrong end of the wand the motion is not as natural.

Now load in that 50 point particle cloud. Look at the list of objects loaded into the LightWaves scene. The wand is first, the null object used to make the motion file is second, finally the point cloud is third. Now moving back to **SPARKS** we setup the *object number* to 3 and the *particle quantity* to 50. Click on the *Track LWMO file* button. A file requester will come up allowing you to select the name of the motion file previously saved. The *Birthrate* control should be highlighted with the rate set to 2. Setup the rest of the controls as you wish. With a low velocity the particles will just drop off the end of the wand. With a higher velocity, a sparkler can be obtained. Render this out for a looksee.

Maybe the particles dropping off are a little to regular for your taste? The *Birthrate* control is a continous flow in accordance with the rate value. This can be randomized by use of the *Clumprate* field. This number is the maximum number of frames that the particles may be "held back" from being released. The particles will be released within this number of frames. They will accrue and stack up if not released. So if the *Birthrate* is 1 and the *Clumprate* is set to 5. Each frame that passes without release adds its particle to the "stack" and when there is a release, all the accumulated particles are lanched. If frame three hits then three particles will be released. If four frames go by before a hit, then four are released. Since 5 is entered in the *Clump* field this is the maximum and no more than 5 frames worth of particles will accumulate. Even finer control can be had by setting the *Birthrate* thru the *Birthrate from list* button. Clicking on this will bring up a requester allowing you to input the exact number of particles to release on exactly what frame!

If you dont want the particles to fall all the way to the ground you can use the *Lifespan* field. This control will end the particles flight on the frame indicated from the frame it is born. We most likely dont want the particles to just stop in the middle of the air, so the *End Behavior* must be *Recycle* or *Kill*, if *stop* is left on *kill* is selected internally. The *Lifespan* button must be selected to allow for the *Lifespan* field to function.

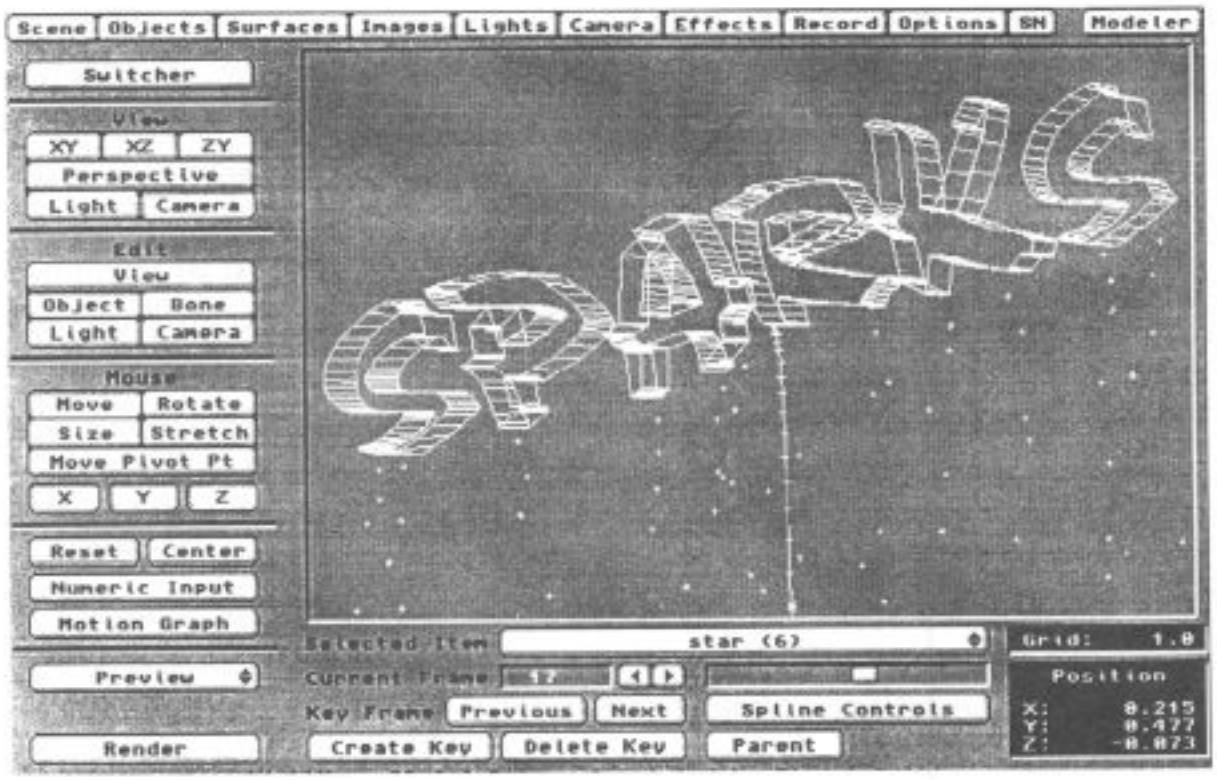
If *Clumping* is off and *Birthrate* is higher than one. You might possibly have noticed that the particles are released in little "puffs" 1/30 of a second apart. This is because they are! They are released at the motion files x, y and z values at the appropriate frame. The faster the motion the more apparent this effect is. We have put in a control to smooth this out called *FrameTween*. This control will "Spread" out the particles released at any given frame between the 1/30 second "time slice" being calculated.

Another control that can influence the fall of the particles is *Inherit Velocity*. This control adds the source motions velocity to the particles velocity so that an abrupt change in direction will actually "fling" the particles off the end of the wand! It is also useful in rocket explosions so that the sparks or whatever are initially traveling the same speed of the object being fired.

3.4 Source Object

Now that we have placed particles into position with the *position* requester and by utilizing a motion file lets look at another very powerful method of placing particles into a initial position. The *Source Object* function allows specific placement of particles and objects by substituting them on the points of a layer in the Modeler.

Let's have a word completely made of points fall to the ground, point by point. From the Modeler select text, type in a word, any word, your choice. This will give you the polygons of the letters. Run the *prick.lwm* macro with 1000 points and your word should have received a fairly dense surface of particles. With the particle objects layer selected. Run the *getpoints.lwm* macro and these positions will be grabbed. Export the particle object model to LightWave and position the camera.

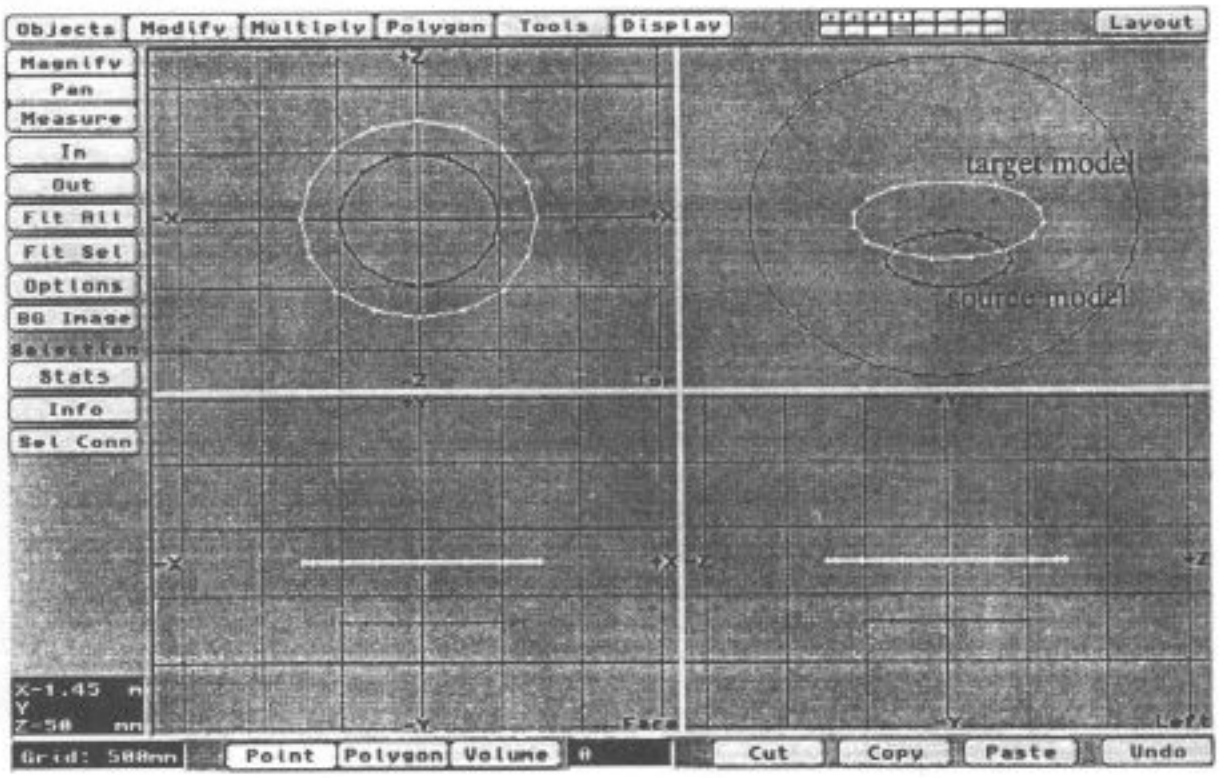


It just so happens that in this case we grabbed the particles positions and then used the same particles for the move. But that doesnt have to be. We can grab any positional model then apply a different number of particles to it. The particles will just "wrap around" and continue at location #1 when they run out of new positions. In other words when 100 particles are used but only 50 particles are in the layer

being grabbed. After they are setup, the Source object will have 2 particles at each vertices location.

From the SPARKS interface select the *Source Object* button. Select *Source* and accept. Enter the *Point Quantity*, if you dont know just look in LightWaves object requester. Setup the *Set Angle* and be sure your velocity is appropriate for the scale of your object. Set the *Birthrate* toggle and rate unless you want all to launch at once. Render it out, compile it and view it.

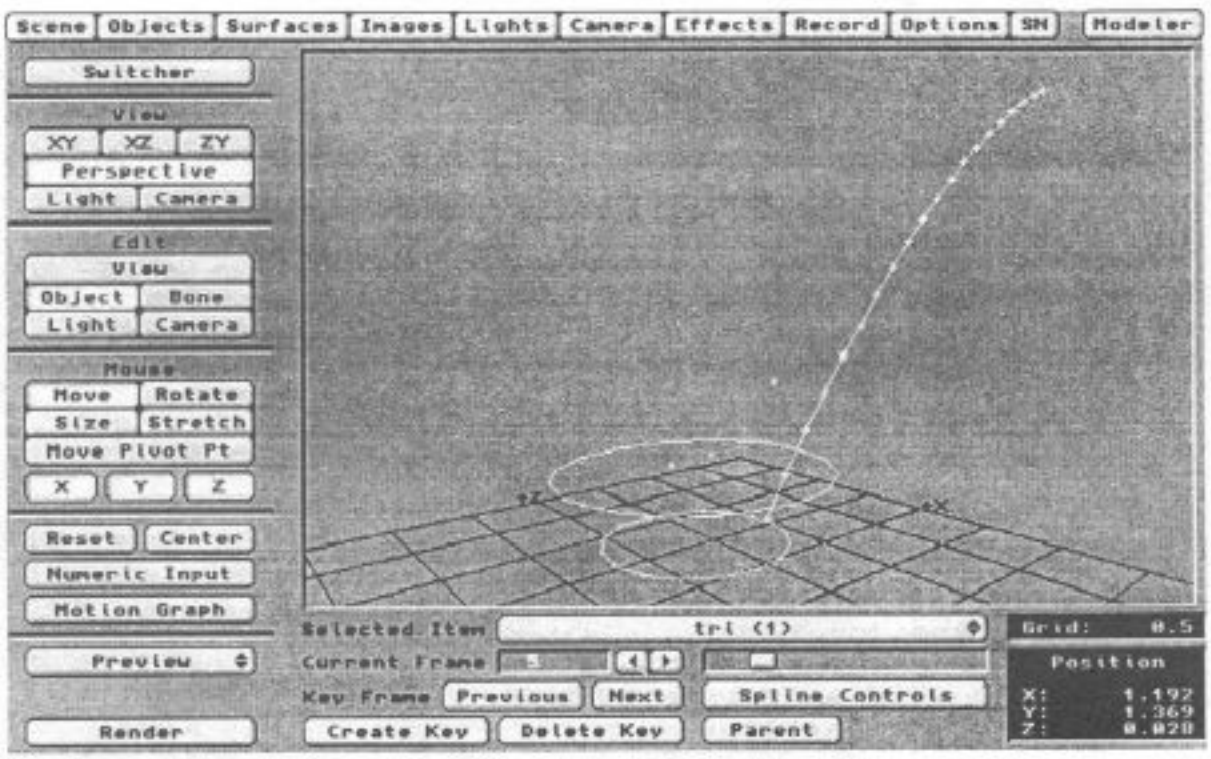
Try some variations such as particles just dripping of the bottoms of your word. This can easily be accomplished by saving the original word you create in Modeler. Then kill all the polygons and cut away all the ones you don't want particles dripping from. Do a merge to eliminate any redundant locations. Run the *getpoints.lwm* macro. to grab all these locations. You can run the points› polys.lwm macro to convert these points into particles. You could also use any other particle cloud should you want more or less particles than locations in your source object.



SPARKS can not only use a source model to position particles but it can also point them with a target model. If we use the *Set Angle* requester to point the particles all particles have the same base angle. This show the limits with that scheme. They can only go different

directions because they are randomized within the *spread* value. Using a model as a target model allows every particle to have a unique angle based on the difference between the source and target position for each particle. This difference describes a vector along which we shall direct each particle. Lets work through a example.

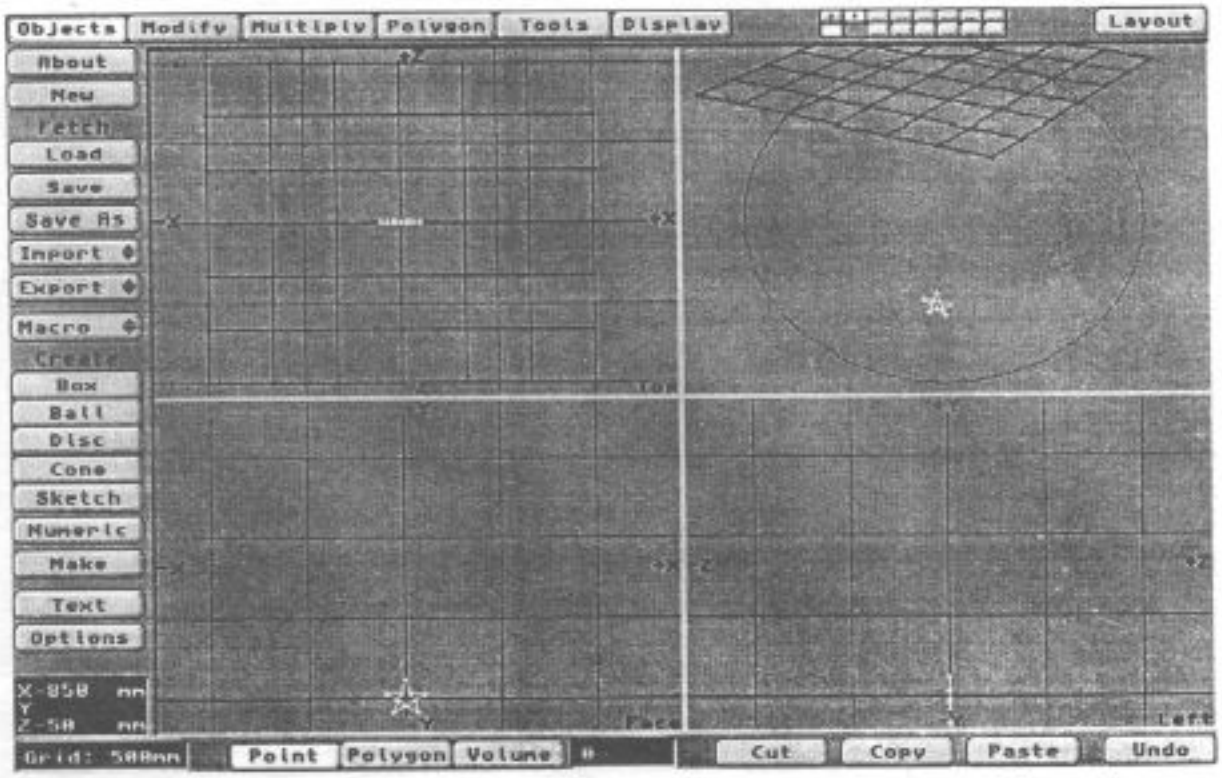
Start in the Modeler by making a disk 1 meter in diameter .5 meters above the y=0 parallel to the ground. Run the *getpoints.lwm* macro and grab this as the source model. Use the scale tool to scale the disk up 150% about the 0,0,0. Again run the getpoint.lwm macro this time selecting *target* from the macro. From SPARKS enter 16 into the *Particle Quantity*. Select *Source Object*, then select *source› target* then accept. Switch *Birthrate* on and enter a *Birthrate* of 1. Set the *End* frame to 30. Set the replacement object to tri.lw in the SPARKS disk2 objects drawer and select save scene. Press *Start*. When done load the scene into SPARKS and observe how each particle is pointed independently of the rest



3.5 StarFall

Let us continue our study by considering how useful it would be to attach an object to a particles path. Now particles can have motion blur and particle blur rendered on them. Now objects can be interspersed with the particles for amazing effects. Watch it though, the memory requirement quickly escalates when you try to load several hundred large objects coupled with the large motion files SPARKS generates.

Lets create 49 falling gold stars. Start in the Modeler and in box create mode. Drag out a square 2 meters square, 2 meters above the 0 line. This should be a box not a cube. The goal is to make a grid parallel to the ground 2 meters above it. Select *numeric* and 6x by 1y by 6z segments. Hit okay and make the box. The box has a total of 49 points. Run the *getpoints.lwm* macro. Run the *points› poly.lwm* macro. Select export and save it to a temporary file as we are only using it so we can see the origon of the move we are about to make.

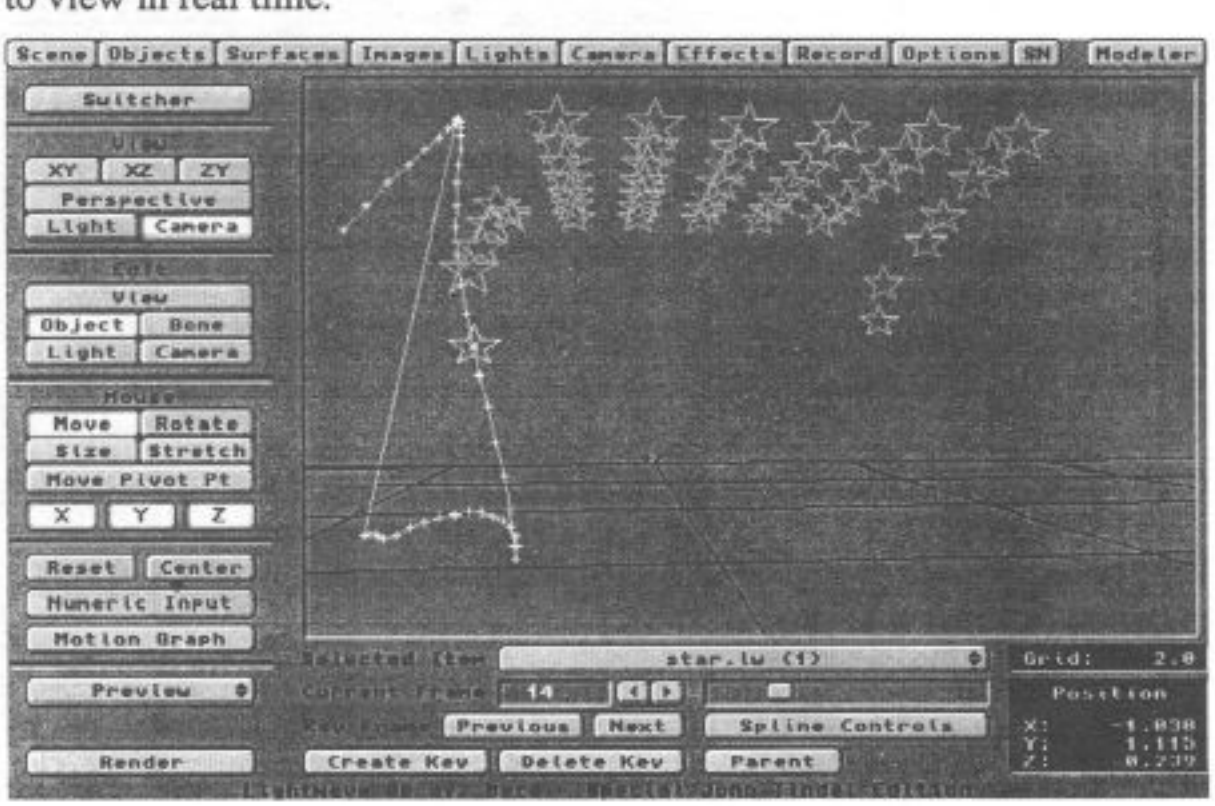


Now switch to a different layer and create the star object. Start by making a disk shape in the front view of 10 points using the cylinder tool. Select every other point and scale them into a five point star shape. Then extrude and bevel(to catch some gleams). Scale it appropriate to the size of the grid so the stars fit between the lines of the grid object on the

background layer. Center the start on 0,0,0, coordinates and save it out.

Now go to LightWave and you should see the grid object. Hot-key back to **SPARKS**. First of all set the point quantity to 49. Set the *Source Object* button to *Source only* and accept. Leave the default spherical direction and set the velocity to .2.

Now for extra action let go to the *Effects* button. This brings up a requester with some forces that can be applied to the particles as they move along their journey. Enter .2 into *Swirl Radius* then accept. Select *Recycle* in the *End Behavior* area. In the pulldown scene menu *Select Objects*. Here is a list of objects to distribute across the source objects vertices. Add the star object to the list by selecting add, then double clicking the star objects file. To delete a wrong entry just highlight it and click delete. Close the window and select save scene from the menu. Choose the file to save the new scene in. Click N*o Render.* After the last frame is calculated the scene file will be built and saved. Now just load the scene file into LightWave and you can make a preview of the move to view in real time.

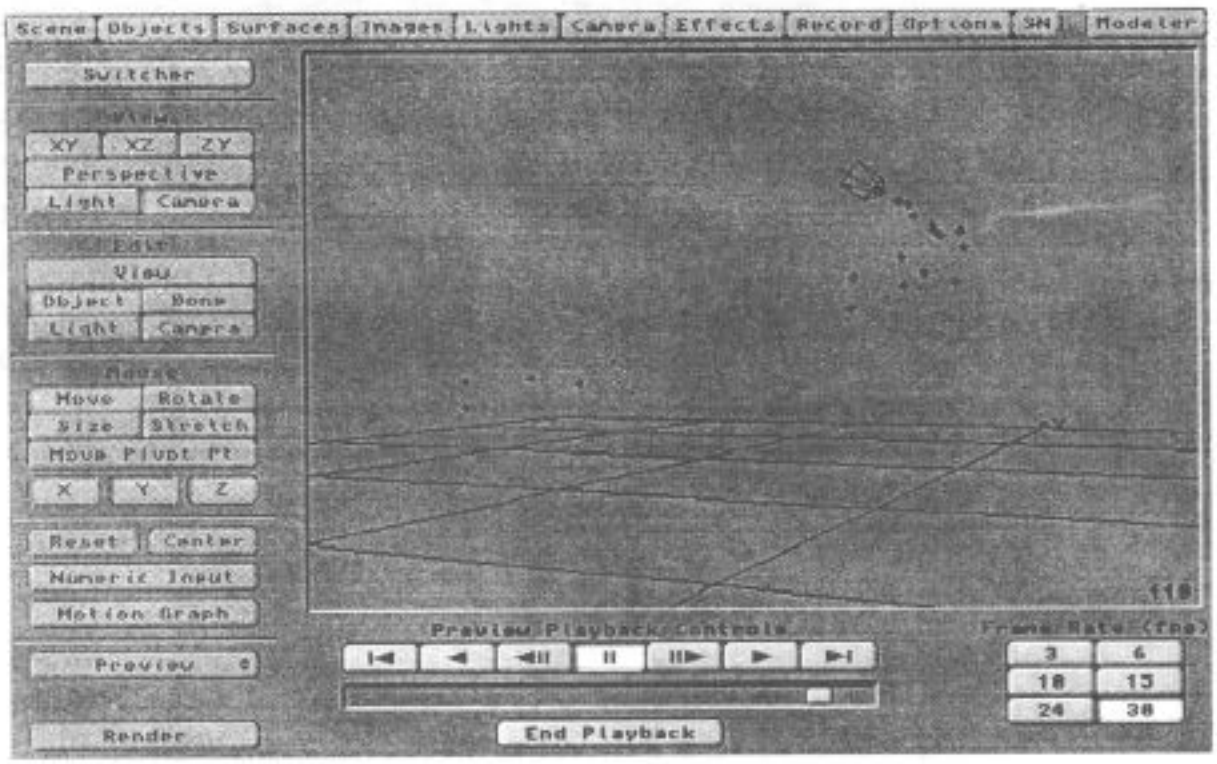


3.6 Nozzles

Lets animate a nozzle object so we can visualize the act of creating a LWMO file to drive to particle flow.

Load the nozzle.lw object from the **SPARKS** disk2 object drawer. The object itself has no effect on the animation but it would be hard to see what is direction we are pointing without it. The arrow shape shows graphically the direction as well as scale.

Set up some keyframes for the nozzle, Moving it as well as rotating the heading and bank. Keep the nozzle above the y=0 because we are going tho leave the default *groundplane*. The flow will emanate from the tip of the nozzle as it is the local 0,0,0 of the object in the direction of the arrow shape.



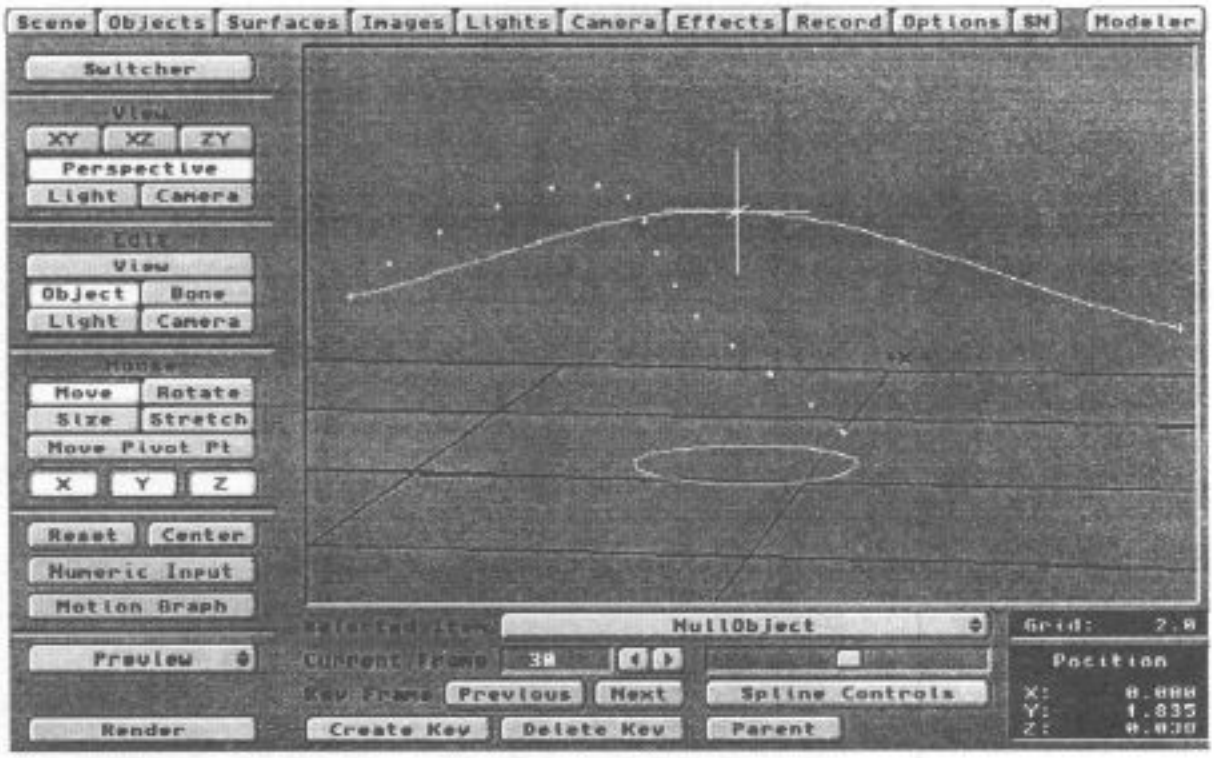
Make keys at 0, 30, 60, 90. Change the scale to .2 at frame 60 and make a key. Make a preview and check the path. With the object selected go to the motion graph and save the motion file to ram. Go to **SPARKS**, or run it if it is not already running. From **SPARKS** change the velocity to 3. In the angle requester change all values to 0. When the heading is 0 the flow is towards the +x axis, as is indicated in the requester legend. This is like a azimuth control. The angle value is like a

elevation control, in that it only points the stream up or down. Between these 2 controls you can specify any point spherically about the origin. By setting them to 0 the flow will point straight forward in the +x.

Set the particle qty to 30, turn recycle on. Select "lifespan" and set the lifespan value to 30 frames. Once a particle is born it will last 30 frames, die, and return to the queue. Set the *Birthrate* on with a rate of 1. Select *Track LWMO file* and a requester will appear allowing you to select the nozzles motion file previously saved to ram. Change the *End* frame to 90. From the scene pulldown menu choose S*elect Object.* Load the tri.lw object from SPARKS disk2 objects drawer. Close the requester. Select save scene from the scene pulldown menu and give the new scene a name. Press start and relax until the scene is finished. Now load the scene into LightWave Using the load *Objects From Scene* button in LightWaves object panel to preserve the nozzle and its motion file. Preview the scene and notice how the *velocity* changes as the scale changed. Go back and change some of SPARKS setting and see how the particle flow is affected. When loading this new scene into LightWave you will have to load over the existing scene thus losing the nozzle and motion. Just load in the nozzle object and load the motion file that was saved to ram: onto it. Another possibility is when you create the scene file select *parent object* from SPARKS scene pulldown menu makes all particle objects parent to a null, which allows you to delete all the children in just one mouse click.

3.7 Targeting

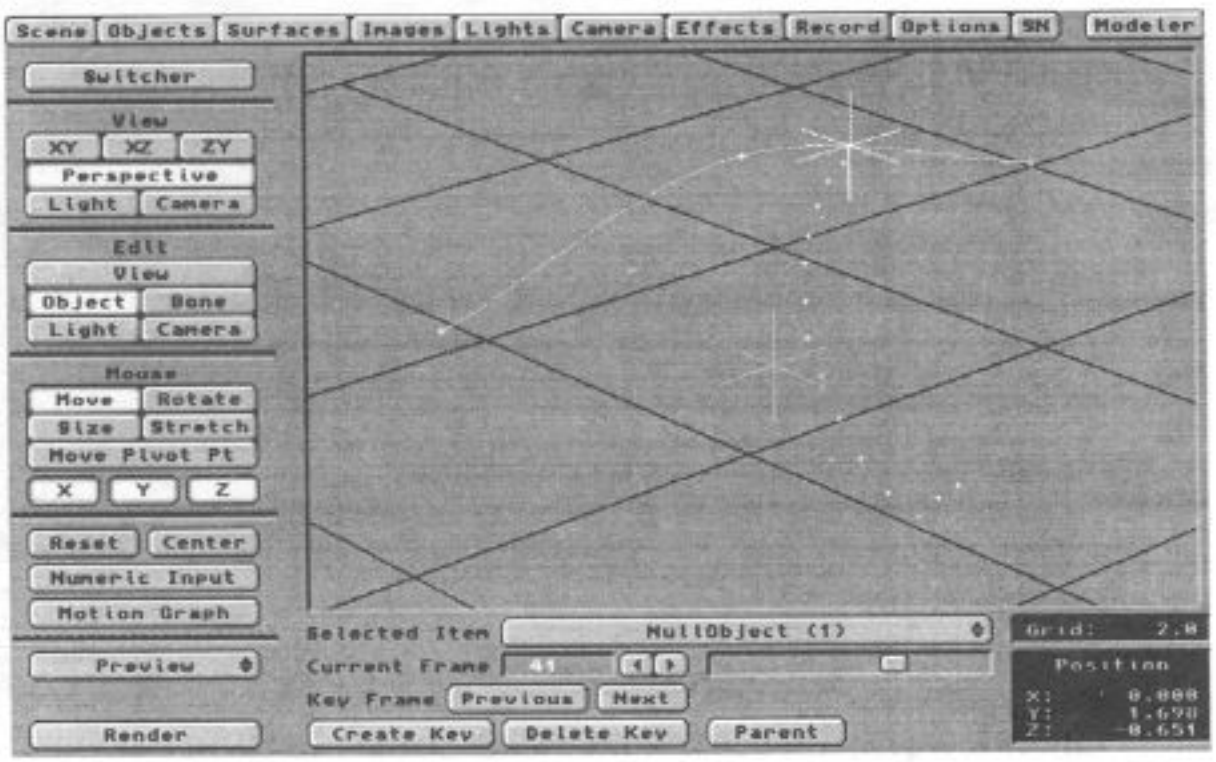
There is yet another way to point the particle stream. Targeting a motion file can shoot particles from the source position toward a moving target. Targeting is applicable to both *Source Object* and *Track LWMO file* methods.



For a *source only* method:

Grab the source positions from Modeler as usual. Make a model using the disk parallel to the ground 1 meter in diameter .5 meters up from y=0. Use the default sides setting gives you 16 points. Grab it with *getpoints.lwm* insuring that *source* is selected in the macro. *Export* the model to LightWave. Add a null object and setup the motion to target and save it to ram. From SPARKS set the *particle quantity* to 32. Select S*ource Object* then *Source Only*. This positions the particles. Selecting *Track LWMO file* brings up a requester asking if you would like to track a object. Select yes and a file requester prompts you for the motion file just saved. SPARKS looks at each particles position when it is born and calculates a direction vector based on the targeted motion.The angle value is ignored because the angle is derived from the target file but the *spread* values are still relevant. If you leave it at the default spherical pattern this example won't look like much, so restrict the *nozzle* to 10° on *heading spread* and *angle spread* (a wide nozzle does not shoot

toward the target consistently enough to illustrate the effect). Set the *Birthrate* to 1 per frame. Set the *Lifespan* on and to 30 frames. Set the E*nd State* to *Recycle*. Select the object to build the scene with as well as the scene to save. Set the end frame to the length of the motion file. In the control menu set *No Move* on. After the scene is finished *Load from scene* will load the particles into LightWave for you to make a preview anim or render out fully.



For a *Track LWMO file* method:

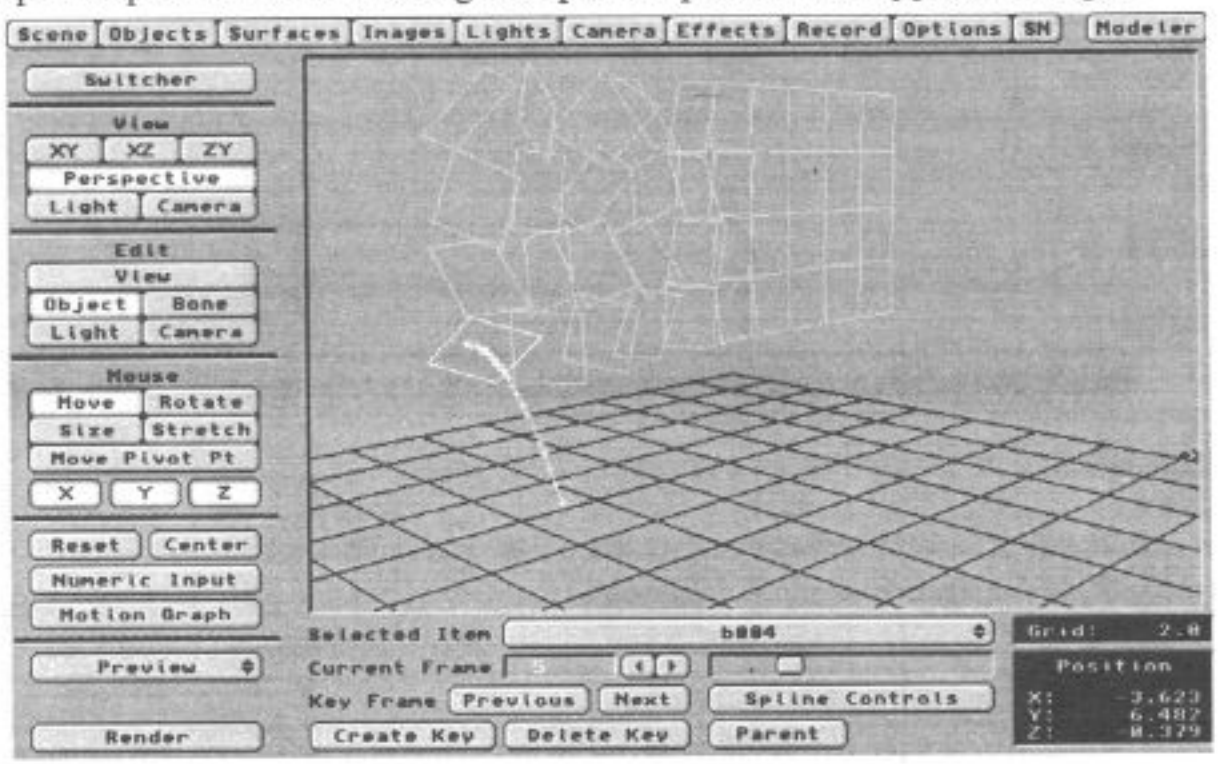
This allows the origin to move as well as point the stream towards a moving target. This of course over-rides the rotational values of the *Track LWMO file* source motion file but not the scaling within it (which will modify the velocity). Simply setup the motion files. Two are required, one for the source of the particles and one indicating the direction to shoot the particles. From **SPARKS** enter the number of particles, say 30. Set the *Birthrate* to 1. Selecting *Track LWMO file* prompts first for the source motion file, then the motion file to target. Set *Recycle* on. Set the *angle requester* to a *spread* of 10° or less for a narrow nozzle . Set the endframe to the length of your motions and the *Lifespan* to 30. Set the replacement object and save scene name. Set *no move* on to prevent calls to LightWave and speed things up a bit. Hit *Start* and load the scene into LightWave when finished for previewing.

3.8 Fragment

In this tutorial we will create a wall of poygons and make it fall away sequentally into its component pieces.

First off we must prepare the model. Make a new directory to hold the pieces of the model that *fragment.lwm* produces. From the Modeler make a vertical plane 10m up from y=0. Go to the numeric and set it to 10x,4y,1z. This gives us a 10 by 4 grid of 40 polygons. Run *fragment.lwm* and set the subdivision to 11y,5y,1z. Select okay. Set the base and path to your new drawer. After the macro runs your model is dissected into polygons grabbed inside the objects bounding volume. The grabs are 1/11 the objects width so any grab can only get one polygon on any grab, hence each polygon is saved individually. When a successful grab is made the fragment is cut and pasted onto a spare layer. The center is calculated, a pivotpoint file is built then the object is saved with a numbered extension. This continues until all the pieces are grabbed and saved. The pivotpoint file is important as it indicates a new center of rotation.

In **SPARKS** set the particle qty to 40. Set the *Birthrate* on and to 1 per frame. Select *Source Object*, then *Source Only* from the requesters options panel. After selecting accept, a requester will appear asking if

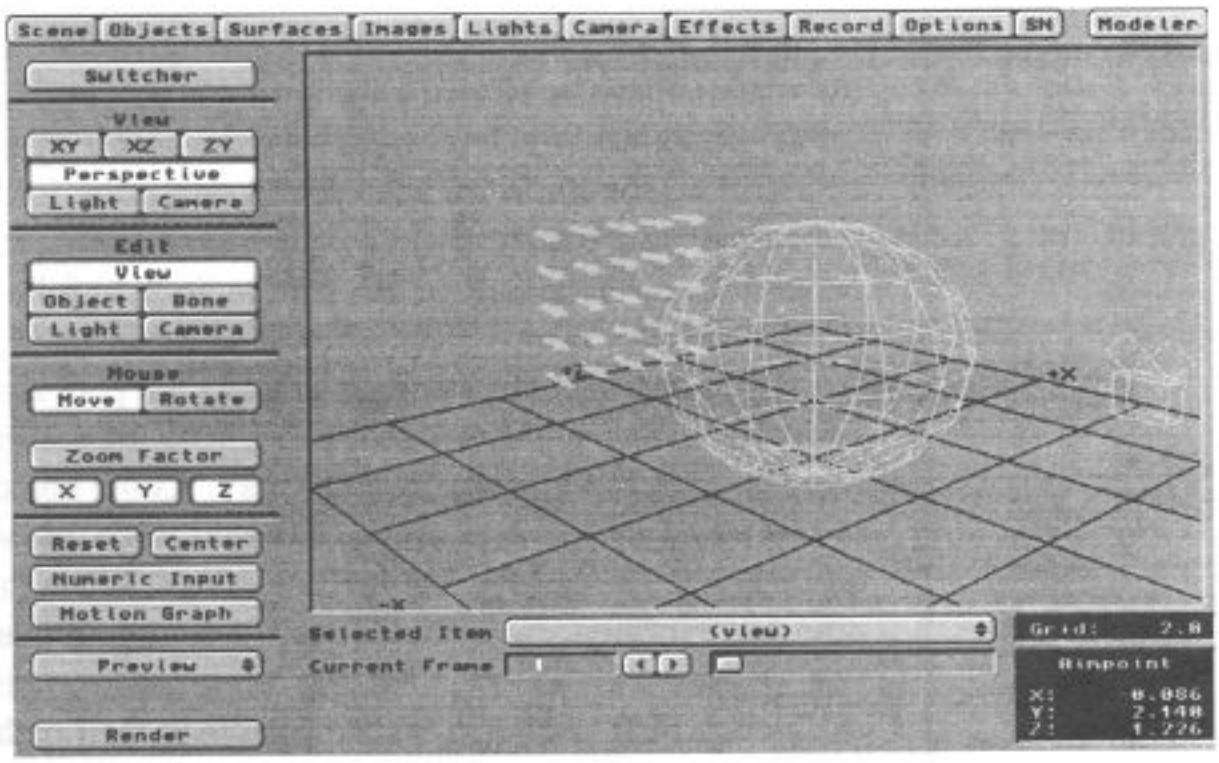


you would like to use the pivotpoint file. Select yes. This requester will only appear if a pivotpoint file is present in t:. From the pulldown scene menu *Select Object*. Holding down the shift key select the *add* button. Pick the drawer that contains the fragments. This will load the entire drawer at once. Change the values in the *evaluate rotation* gadgets to "r". This will make the fragments randomly rotate as they fall. Select *Save Scene* and give it a name. Set *No Move* from the control pulldown menu. Because we are not using a target to point the particles, we'll use the *angle requester* as usual. Set it to a small spread say 10 or 20° and shoot it toward the -z at heading 270°. Set a low velocity like 1m/s. When done load the scene in to LightWave and preview it. See how the effect progresses from left to right down the x axis. This can be changed by setting the different parameters in the fragment macro. The effect can be ordered down any axis or can be random. When the fragment macro is run a target file is also produced. The relationship between the center of explosion and each fragments center describes a vector along which the fragment are initially directed. **SPARKS** use of source and target files is transparent to the user. They are temp files placed in t:.

3.9 Flocking

Flocking is where you setup a path and have many objects follow it with out having to actually set up a lot of individual motion paths. Flocking a group of objects is not hard at all. For starters we need a motion file of the path that the objects are going to follow. Just as in the traveling origin tutorial for the wand, we need a LightWave motion file to indicate the path. Save this motion file.

Next we need to setup the origin of each flock object by going to the Modeler and placing a point where the individual flock objects should be. We could also just make a primitive, maybe distort it or whatever, then grab the points using the getpoints.lwm macro that has been provided. Now exit Modeler to LightWave. Hot key to **SPARKS** and select the flock button. The data file will be read in and you will be presented with a requester defining some spring settings for the flock. These spring values affect the behavior of the objects as they move along the path. The general idea is that a spring force holds each object in its home position along the path. The objects can be nudged out of this home position by external forces such as a gravity well. Yet when the restoration force of the spring overcomes the external force opposing it, it will return to its home position at a rate determined by the settings.

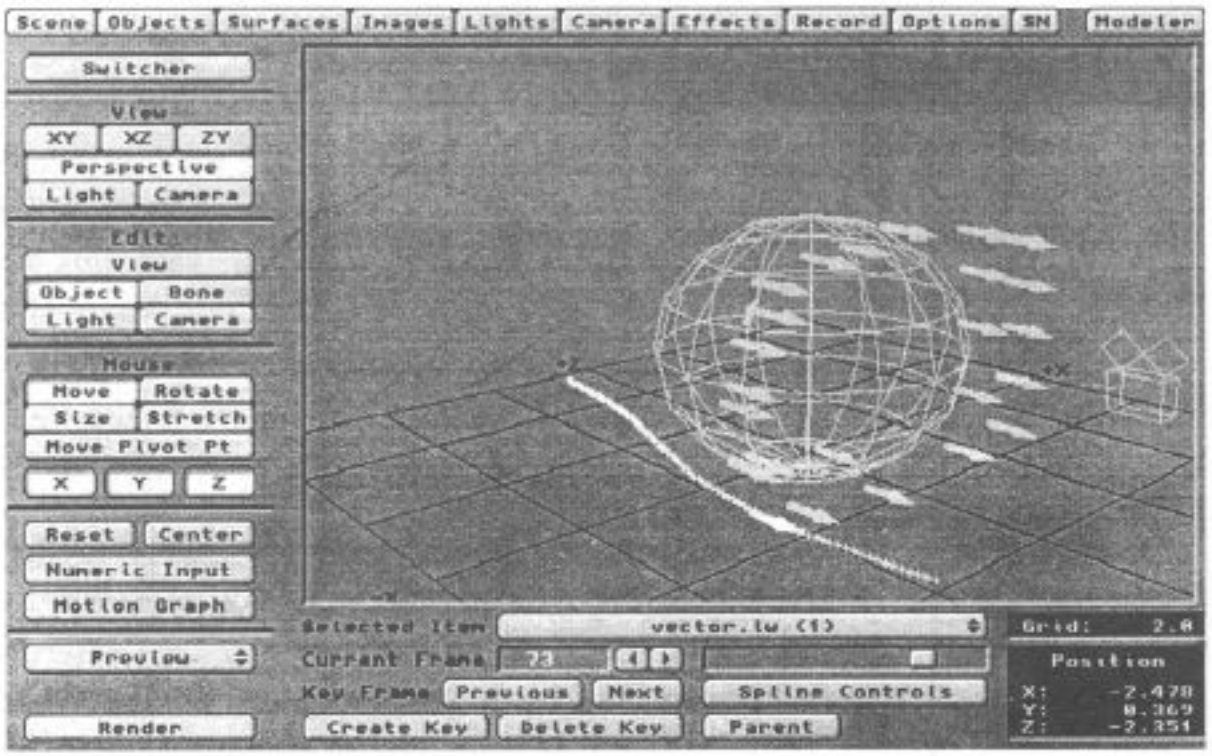


These settings include:

Stiffness which controls how much force is required to push the object from home.

Restoration controls how quickly a object is brought home.

Damping stops the oscillations that a spring has as it moves past its center point slows, stops and accelerates toward the center. It functions the same as a shock absorber on your car.

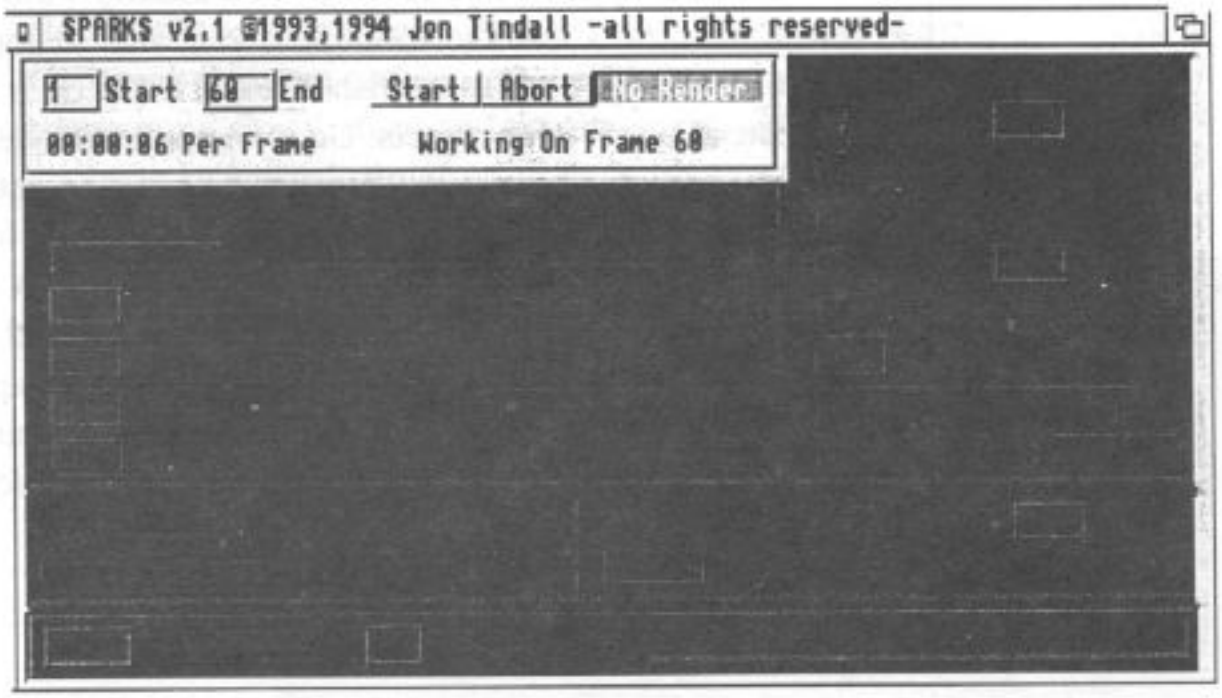


To help you visualize these settings click on the gui button and the screen opens up showing the spring explorer. This allows you to tug on the small box which is attached to the center by a spring. The springy behavior can be studied and used as a start point for the settings. After the spring has been setup close the window and a requester will come up for you to select the motion file for the flock to follow.

Your flock is set to follow along the path alright but for a real cool effect setup a gravity well or two with the antigravity button on for a repulsive force near or in the paths way. As the objects cruise along their merry way they will encounter a external force that will distort the flock according to the gravity and spring settings. When the flock is past the disruptive influence it will pull itself back together unless the spring restore setting is too low. This effect is great for parting a school of fish or flock of birds as they avoid a obstacle in their path.

Now what is left pertains to the scene file that will be built. Go to the menu and choose select object. A requester allows for the multiple selection of objects for your flock. You usually want to align to path so select that now. Close the window. If you want to apply a displacement map to your objects to make fish swim or birds fly it can be set up now and placed on all your objects. Go to the menu labeled appropriately enough setup displacement. This opens a panel to enter all the mapping information. One addition is the small r button next to x, y and z center fields. This button will randomize the field next to it when toggled on. This offsets the map so the phase is different for each object. If this is not done all your objects will be moving (swimming, flapping) in unison. Choose the save scene menu item and select a save name for the scene. Now set the start and end fields and start. Load the scene into LightWave when it is done and WOW!.

4 Reference



Start and End Field

These two fields control what frames that will be rendered. If your script calls for the action to begin at frame 220 and run to frame 500, enter 220 in the *Start* field and 550 in the field labeled *End.*

Start

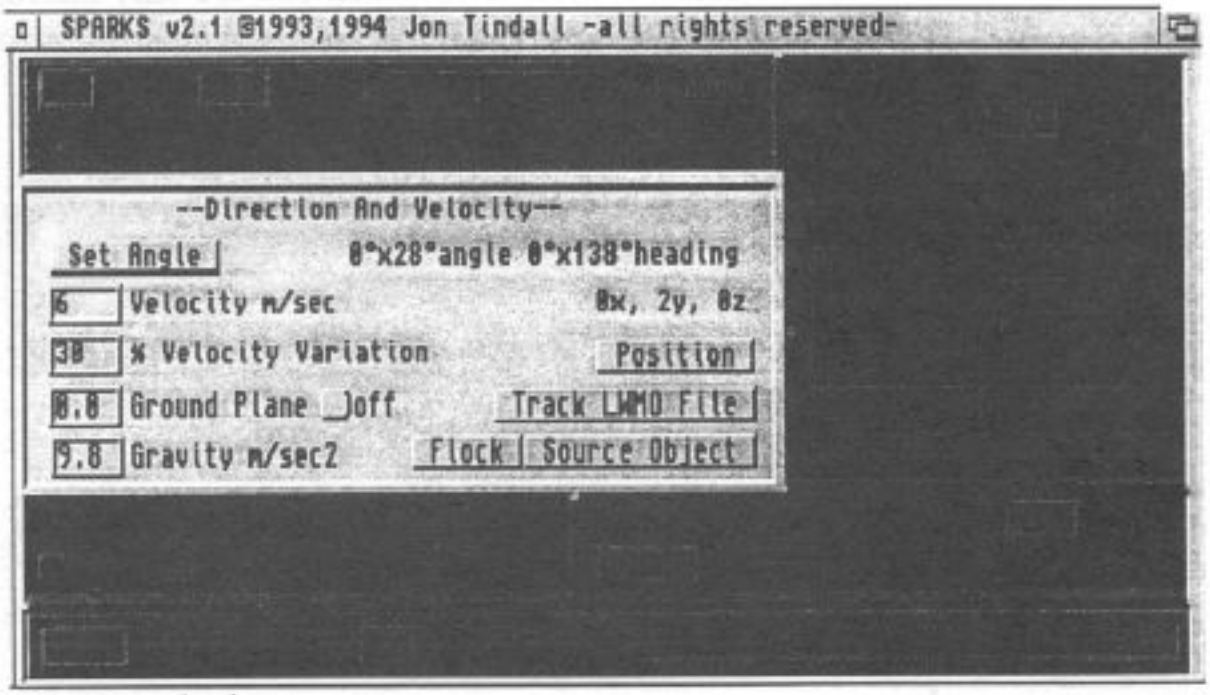
The *Start* button begins the calculations starting at the *Start* field value and ends at the *End* field value.

Abort

Aborts the calculation in progress. If you are in LightWave, rendering don't like the way the particles are moving just hit escape. Then go to **SPARKS** and click on *Abort.* After the scene is Aborted, the data in memory is flushed. This means that after you Abort a scene there is no way to continue calculating the simulation, you will have to start over from the beginning.

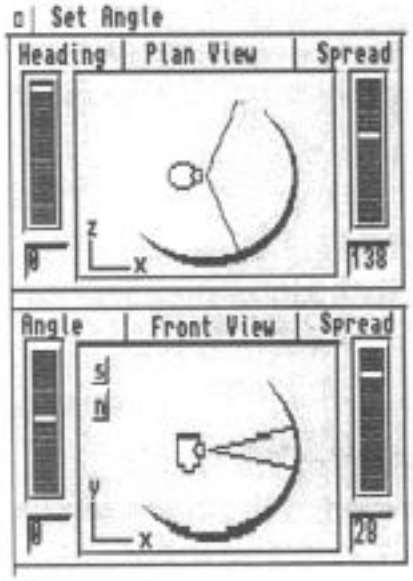
be instructed to render the frame. This is of use to see where the particles are going, by Hot-keying back to the Lightwave interface and selecting edit object. It is also useful when you are only interested in the scene file that is going to be saved out after the simulation is run.

4.2 Direction and Velocity Controls



Set Angle button

The Set Angle button is to select a angle for the particle simulation. By positioning the front view(lower gadget) gadget the user is able to select a angle and width of the stream. By selecting the plan view(upper gadget) gadget the user is able control the heading and width that the front view values will be swept out through. In other words the lower gadget sets the angle up-down, or y coordinate for the stream. The upper gadget sets the direction north, east,south or west, or for x and z coordinates. This "nozzle" can also spray in a ring like a sprinkler or a full spherical spray. These settings are ignored when using the source to target model as this method bases the direction on the difference from source to target models.



Velocity Field

The velocity in meters per second that the particles are projected from either the "nozzle" or from the *Source* and *Source › Target* methods. This is the initial velocity only and subject to external forces as

well as the velocity variation control. The *Velocity* setting is ignored when using the source to target velocity model, and when you are flocking, as these have their own method of determining velocity.

Velocity Variation

The *Velocity Variation* control adds and subtracts a number no higher than the indicated percentage from the initial velocity. This value will apply anytime the *Velocity* control is applicable.

Ground Plane

Enter the *Ground Plane* to setup the ground position for bouncing on or setting the *End State*. This is equivalent to the Y coordinate number in LightWave. This number can be negative.

Gravity

On Earth the gravity is 9.8 meters per second squared, on your planet it may be different. This number affects how quickly things fall and pickup acceleration. If the number is negative things fall up! Useful for bubbles and balloons and such. If the *gravity* is 0 things just float subject to external forces.

Position

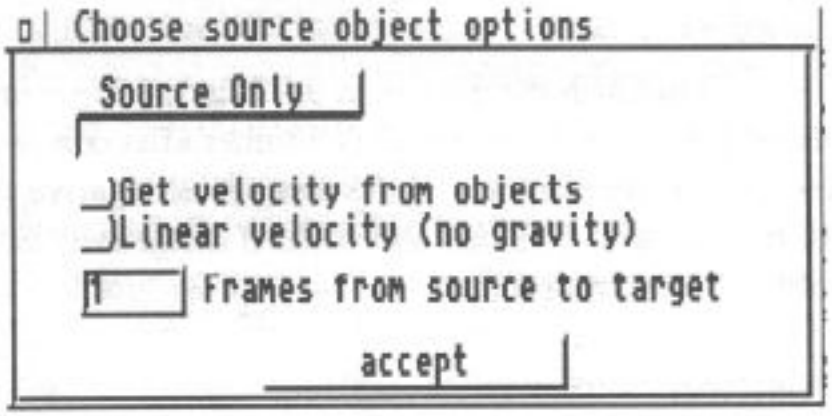
Press Position to bring up a requester to enter the stationary position of your "nozzle". This position can be anywhere above the ground plane.

Track LWMO File Button

Track LWMO File enable the "nozzle" position to be moved by a motion file saved out from LightWave. This motion file or envelope can be built by moving a null object to the position that the particles are to originate. When that is done bring up the motion panel and save the motion file to disk. When *Track LWMO File* is clicked on a file requester comes up and allows you to choose the previously prepared file.

Source Object

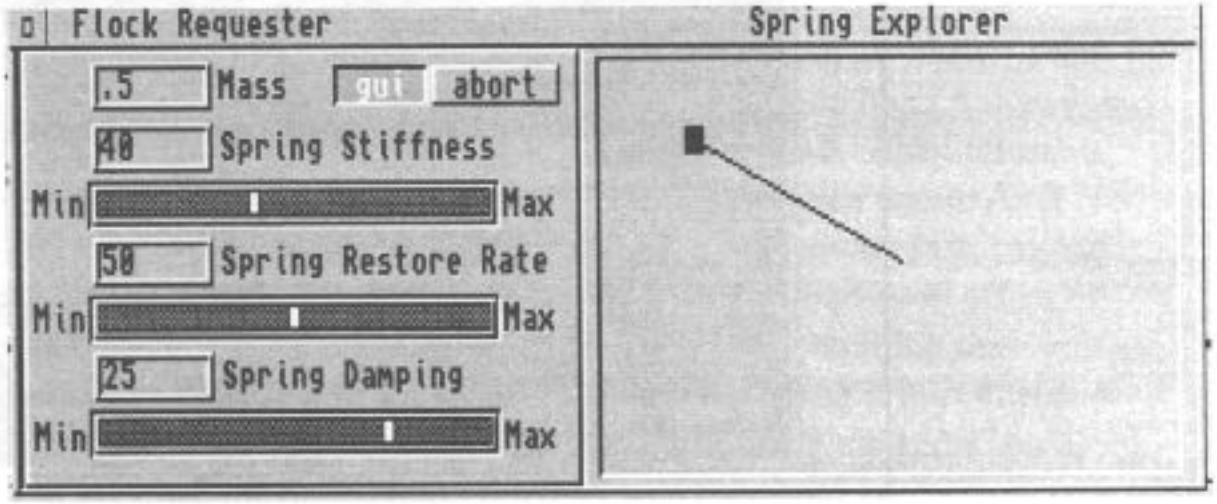
Source Object will bring up a requester with some additional choices. The source object is used to setup the initial starting positions of the particles for the simulation. The model whose vertices are to be used as starting positions is loaded into a layer of the Modeler. The *getpoints.lwm* macro is run. Then go back to Lightwave. Upon acceptance the positions are read in and the particles are distributed across the source models vertices locations. If there are more particles than locations they will just loop across the model leaving more than one particle at each vertices. A target position may also be grabbed. Which will allow setting the direction only (Gravity method), or direction and velocity can be extrapolated (Velocity method). When using the target velocity method the velocity is calculated by the distance from source to target over the duration fields frames.



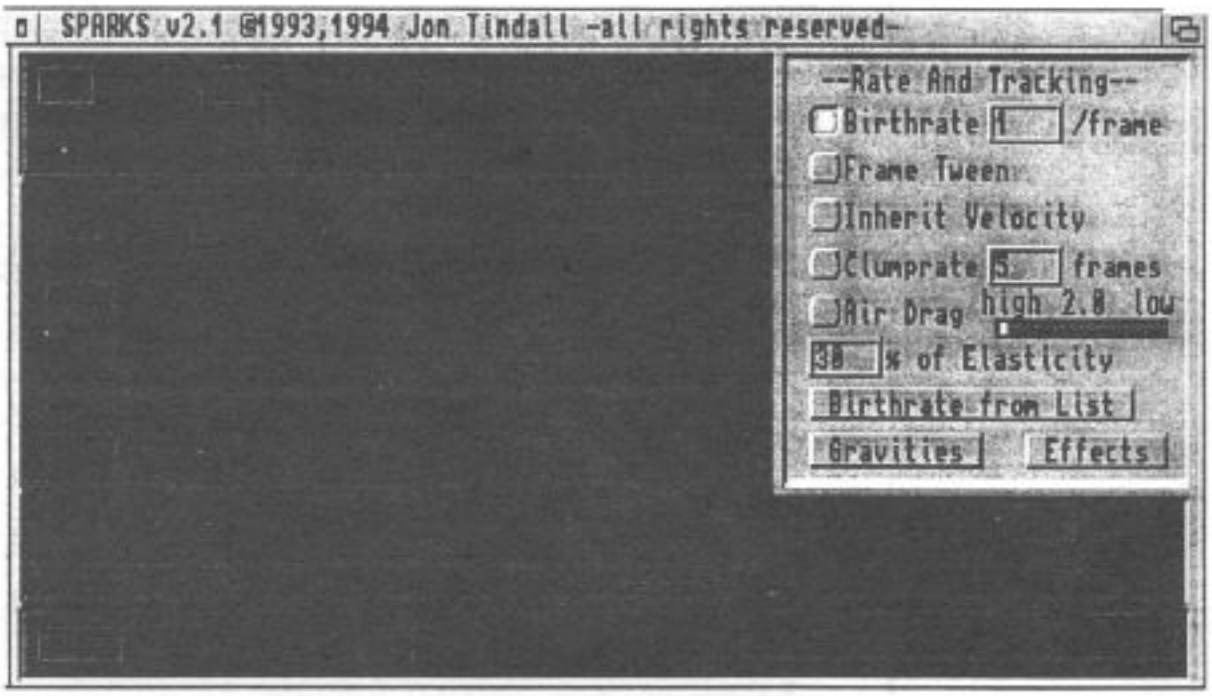
Flock

Flock will allow you to send a group of particles whose positions are derived from a source model, down a Lightave path. First have both a source model whose vertices positions have been grabbed and a LightWave motion file ready. The motion file must have a key a every frame that you intend the scene file to last. The motion file sets up a home path for all particles in the flock. Although external forces may move the particles from their home path, the spring settings allow the particles to return to home when the restorative force overcomes the external force. When *Flock* is selected a window comes up for you to enter the spring stiffness 99=stiff, 1=soft, as well as spring damping 100=undamped(hardly ever stops), 0=totally damped(comes back without much rebound). The restore rate is the amount of restorative force applied to counteract the external forces+spring forces moving the particle from its home path, 99=high restore(most energy is returned),

a handle for a spring that you can pull away from center and release. The spring is running in real-time calculating the same formula applied to the flocking objects along their path. Abort will stop the calculation. Clicking *gui* again will close the Spring Explorer portion of the window. Close the *Flock Requester* window and a file requester will come up allowing you to select the motion file previously prepared for the flock to follow.



4.3 Rate and Tracking



Birthrate

Birthrate controls if all particles fire at the first frame all at once or are launched in a continuous manner of x per frame. The number in the field indicates the quantity of particles to shoot at any given frame. 5 will launch 5 per frame, .5 will launch 1 ever other frame and .1 will launch 1 every ten frames.

Frame Tween

Frame Tween is only applicable when the *Birthrate* control is on and set higher than 1 per frame and you are tracking a moving path via *Track LWMO File* button. What it does in essence is spread the individual particles out between frames. When a fast moving path is being followed each 1/30 second is a discrete slice of time. Its like a puff of particles being let go at each 1/30 second interval. This slices the distance between frames by the quantity of particles being let go and spreads them between origin locations. A neat side effect of this control is that you can purposely make a extremely fast path, say from one side of your screen to the other in one frame and feed that path to *Track LWMO File*. If that path goes from one side of the screen at frame 1 to the other side a frame 2 and you launch 100 particles all at once. They will be "tweened" that is spread out the entire distance between the two

keys, instead of all being let go at the frame 1 origin. This can allow you to create "curtains" of falling particles and objects, as well as just smoothing the release of particles.

Inherit Velocity

When *Inherit Velocity* is highlighted the origins velocity is added to the velocity on the frame it is released. This is only going to be of use to you if you are tracking a moving origin via the *Track LWMO File* button. Inherit Velocity is useful when making sparks fall from a wand. As it makes the sparks fling of the end of the stick if it is moving fast, instead of just pour off. It can also be useful when making a explosion effect on a moving ship. By making a few keys with a null and tracking the explosion site, saving the motion envelope, and bringing it into *Track LWMO File*, your explosions velocity will be traveling the same initial speed as the craft thus enhancing the reality of the shot.

Clumprate

The *Clumprate* add a randomness to the particle *Birthrate*. The rate field represents the maximum quantity of frames to hold back the particle flow before releasing them. The actual number of frames that will be held back is determined randomly. The number of launchless frames will not exceed the maximum Set in this field.

Air Drag

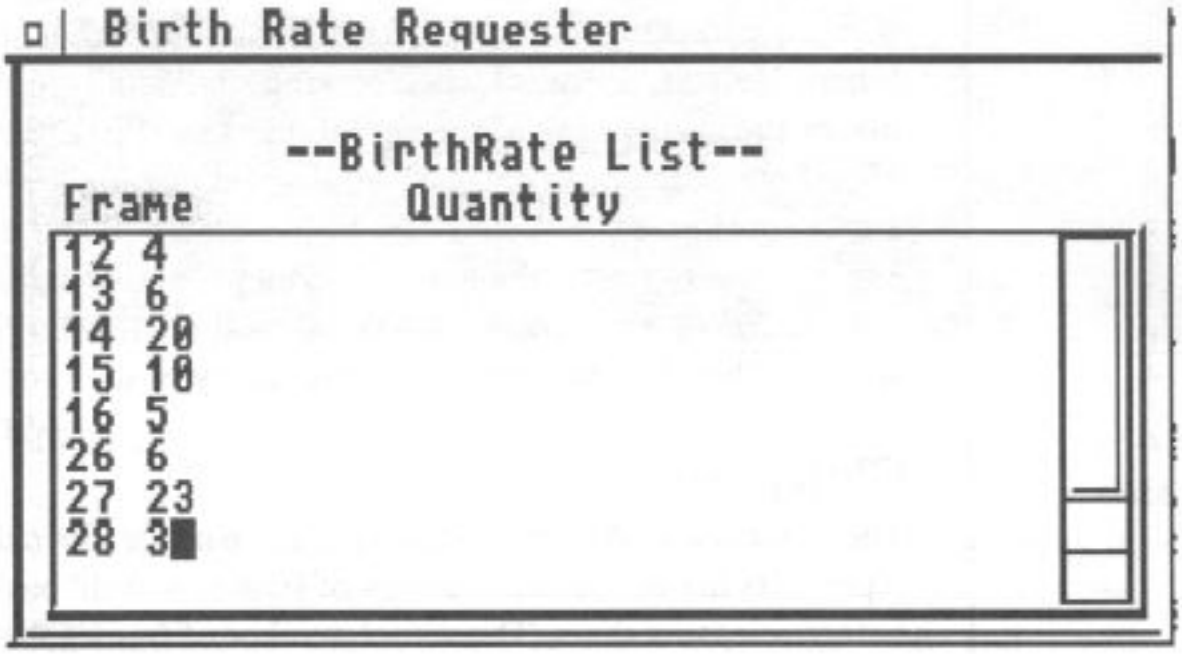
A simulation without air drag is like tossing marbles around. This maybe fine for somethings but for snowflakes and paper wadded up you need *Air Drag*. This control makes a particles velocity rapidly fall off into a terminal velocity. Normal values will be in the range of 1 to 3.

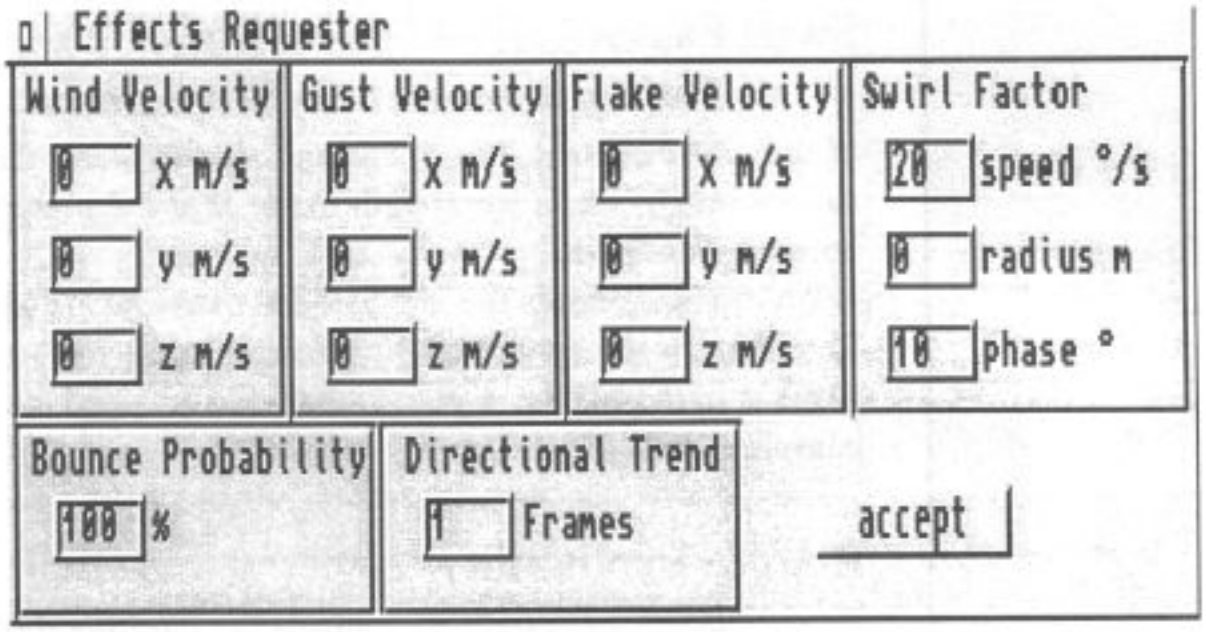
Elasticity

Elasticity controls the amount of energy returned to the particle at each contact to the ground plane. At 0% all energy would be lost and the particle will not bounce. At 100% all energy will be returned and it will bounce indefinitely. The default setting of 30 returns 30% of the energy arriving at the ground plane. This is a good setting to start with.

Birthrate From List

Sometimes a continuous or pulsed birthrate just doesn't cut it. You need to indicate exactly 20 particles are born on frame 6 and 35 on frame 17. *Birthrate From List* allows just that. Clicking on the button brings up a editor where you simply enter the frame number that particles should be born, a space and the required quantity of particles. Very handy for explosions where the starts are positioned by *Track LWMO*, and the *Birthrate* and *Quantity* are controlled by *Birthrate From List.*





Effects

The *Effects* button when clicked will bring up the *Effects* panel. The *Effects* panel has velocity settings for *Wind, Gusting, Flaking* and *Swirling*, as well as controls for *Bounce Probability* and *Directional Trend*. These Forces are additive and can all be on if that's what you desire. They can be used with all modes and effects such as flocking. Lets look at these one at a time:

Wind

Wind is constant. It blows on all particles evenly. It is mainly useful for blowing a plume of a fountain to the side. The number is input in meters per second for each axis.

Gusting

Gusting is periodic. It blows on all particles evenly. It does change direction and intensity not to exceed the number input. It is affected by the *Directional Trend* field. The number is input in meters per second for each axis. Severe Gusting in the Y can lead to particles that can't seem to land.

Flaking

Flaking is periodic. It does not blow on all particles evenly. It is calculated for each particle individually. It changes direction and intensity not to exceed the number input. It is affected by the *Directional Trend* field. The number is input in meters per second for each axis. This control is nice for snow and such where the gust is better suited for rain.

Swirl Factor

Swirling is periodic. It can be synchronized or staggered by use of the *phase* control. The *Speed* and *Radius* affect the diameter of the swirl which is like a spiral path in the X and Z plane traced around the normal path down.

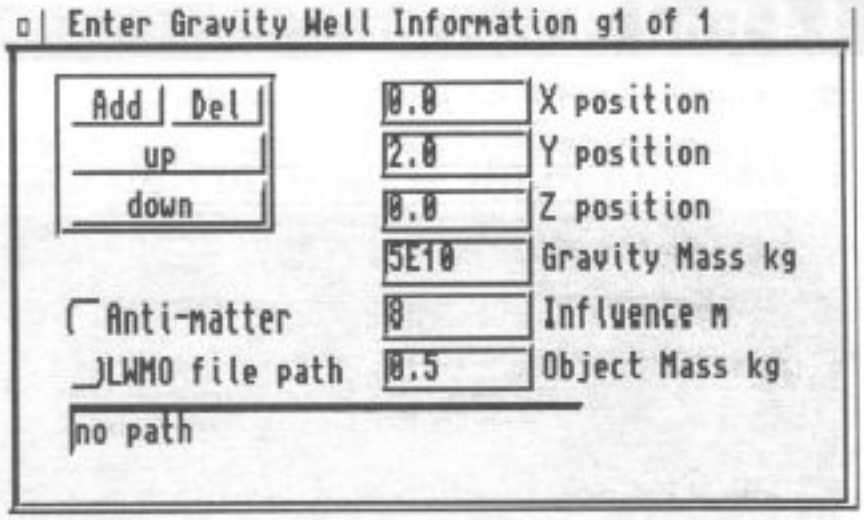
The *Speed* is the angle added to the sin() function for each frame. The *phase* is the angle that the particles are offset from each other. The angle is in degrees so if you entered 12 for speed the spiral would go a complete 360˚ in 30 frames.

Bounce Probability

Bounce Probability controls the likelihood of a particle bouncing when it hits the floor. At 0 no particles will bounce. at 100 they all will provided that they have enough energy to exceed the *velocity threshold.*

Directional trend

Directional trend controls the likelihood of a particle changing direction. The lower it is the more often a directional change will occur, the particles movement will move more chaotic. The higher it is the less likely a directional trend will be reversed this would appear less chaotic.

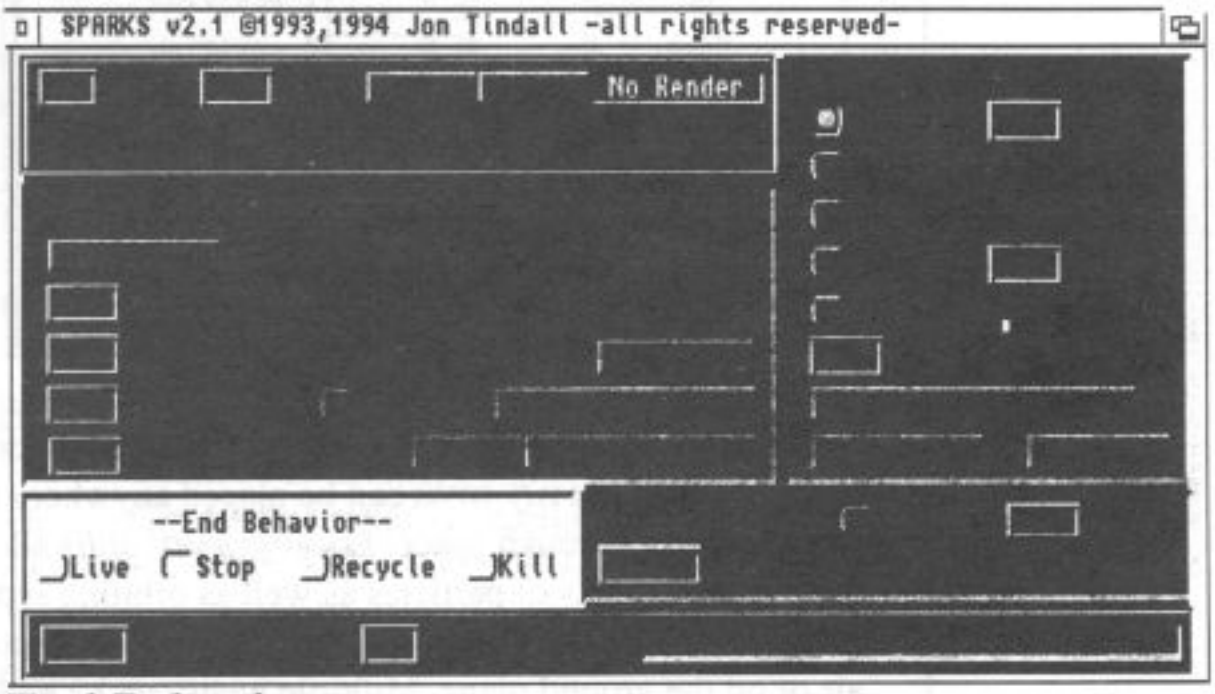


Gravities

The Gravities button accesses a requester for local *gravity well* control. Local *gravity wells* can be placed anywhere as well as be given a path to follow through space. Local *gravity wells* allow for bending streams and flocks of particles as well as orbiting type affects. Local gravity wells on a path can suck and push around active particles in the air or on the ground plane. Multiple wells can be added and can be positive or negative. The fields for the X, Y, and Z position allow positioning.

The Gravity Mass is the *gravity wells* mass in kilograms. This must be a very large number. It is such a large number that it must be expressed in exponential form. This is a number followed by a capital E the the number of zeros that follow. So a object with the mass of the moon would be expressed by the number 7.4E22 That is 740,000,000,000,000,000,000,000 kilograms. So you can see to avoid typing in all those zeros, exponents are great. Now this number is going to vary depending on the mass of the particle and the radius of influence, as well as what it is that you are trying to achieve. but if you start at the defaults and adjust from there you should have no problem. The *Radius Of Influence* will let you wall off the gravity from affecting the flow until it get near enough. The gravity mass is scaled to fit within this radius. So with a smaller gravity mass scaled to the radius gives you a soft edged force great for pushing thing arounds. The particles mass can be set here also. The particle mass has great effect on the attraction after all gravity and mass are interlinked . Less mass produces less gravity effect. The path button brings up a requester allowing the selection of a previously saved LightWave motion envelope for the gravity to follow.

4.4 End Behavior



End Behavior

The *End Behavior* is 4 mutually exclusive states that affect what happens to a particle when certain conditions are met. The particles have 2 states, active and inactive. These states are important to how the particles get updated when they run out of steam or are just hanging around waiting to be launched. It is also important when you are intending to save a scene file out with a fade envelope as the fade envelopes are built depending on the state of each particle. When a particle is inactive, it is updated positionally each frame depending on the method used. This makes sure that they follow a moving origin. When a particle is active, it is on its own. It runs through the simulation until it reaches a condition that allows you to take control again. They are:

Live

Particles stop yet stay active. This is useful for starting out the simulation with particles laying on the *Ground Plane*, a condition that would normally set the state to inactive. And when they are inactive no gusts can blow on them. This would also allow a moving anti-gravity well to plow into them and blast them around.

Stop

Particles are switched inactive when on the ground and their velocity is under the velocity threshold value. Stop means when their state goes inactive they will stay where where they lay.

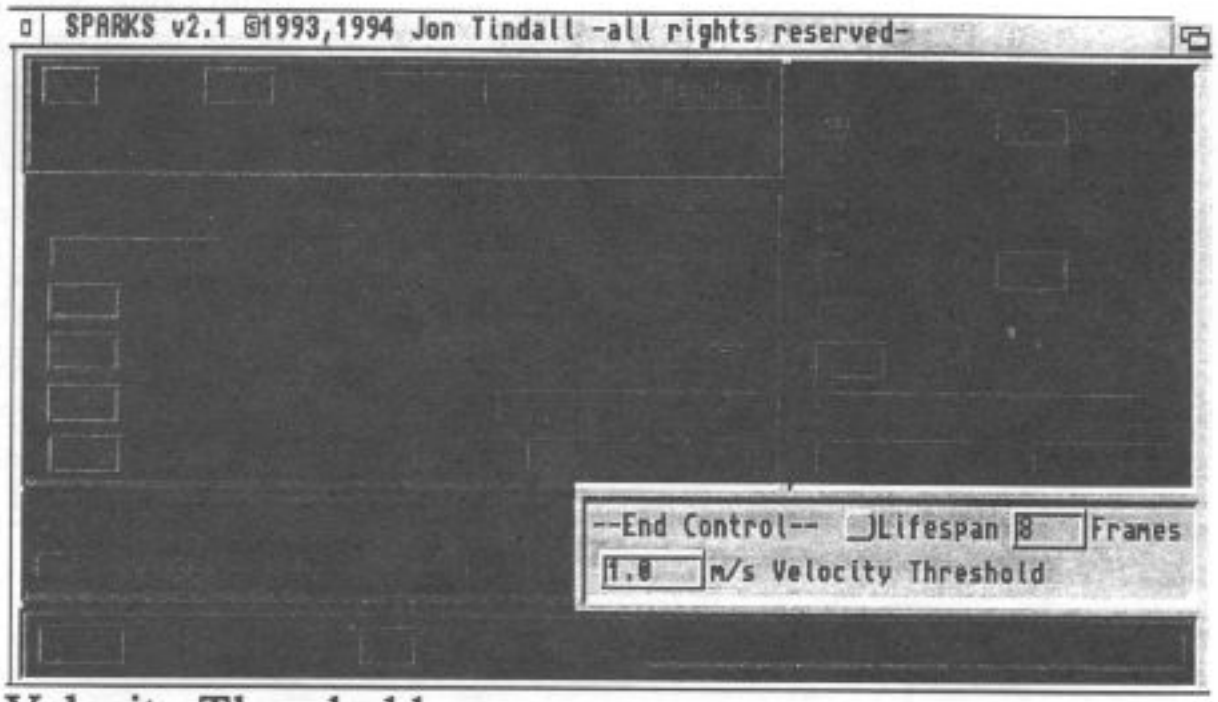
Recycle

When the particles are switched inactive they will be snapped back to the current origin in one frame. Depending on the other control factors, when the flow of particles runs out it will start recycling particles if they are available. It will not steal active particles to use. This means that if you "run" out of particles the flow will stop and continue as the particles are recycled back into the inactive state and sent to the current origin

Kill

When the end state is reached the particles will return to the current origin in one frame. There they will stay inactive, never to be launched again.

4.5 End Control



Velocity Threshold

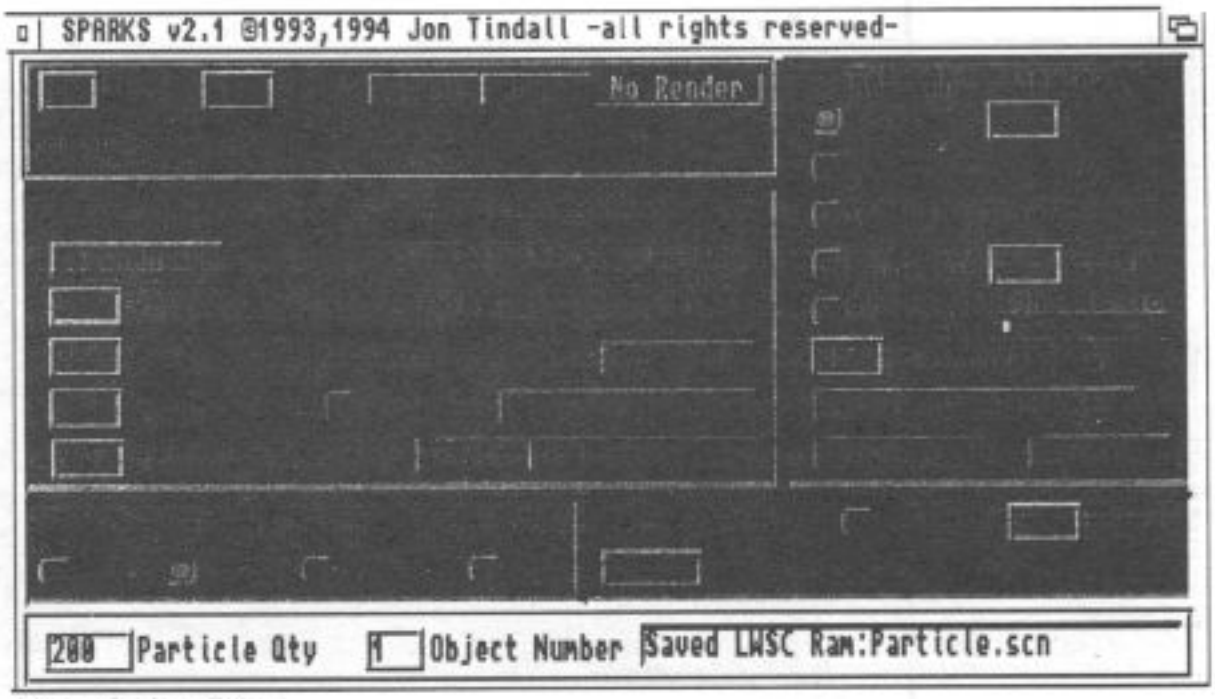
The *Velocity Threshold* affects when an particles endstate is switched to off. When a particles velocity drops below this value while it is in contact with the ground plane its *End State* is switched. The reason for a ground plane qualifier is that you would'nt want your particles to stop in mid air would you? You do? Well then you might want to use Lifespan.

What if the *Ground Plane* is turned off? Your particles might fall forever. Instead we made the velocity in this case, the velocity that when exceeded switches the particles endstate to off. Think of a particle falling and picking up speed as it does. When its speed exceeds the *Velocity Threshold* it would be recycled or killed. It can't be stopped because that would leave it hanging in mid-air.

Lifespan

Lifespan sets the end state at the end of a specific lifespan. If you wanted each particle to last for 10 frames, set this field to 10 and toggle on *Lifespan.*

4.6 Particle Control



Particle Qty

Enter the number if particles to move. If you are saving a scene this is the number of objects that will be in the scene as well.

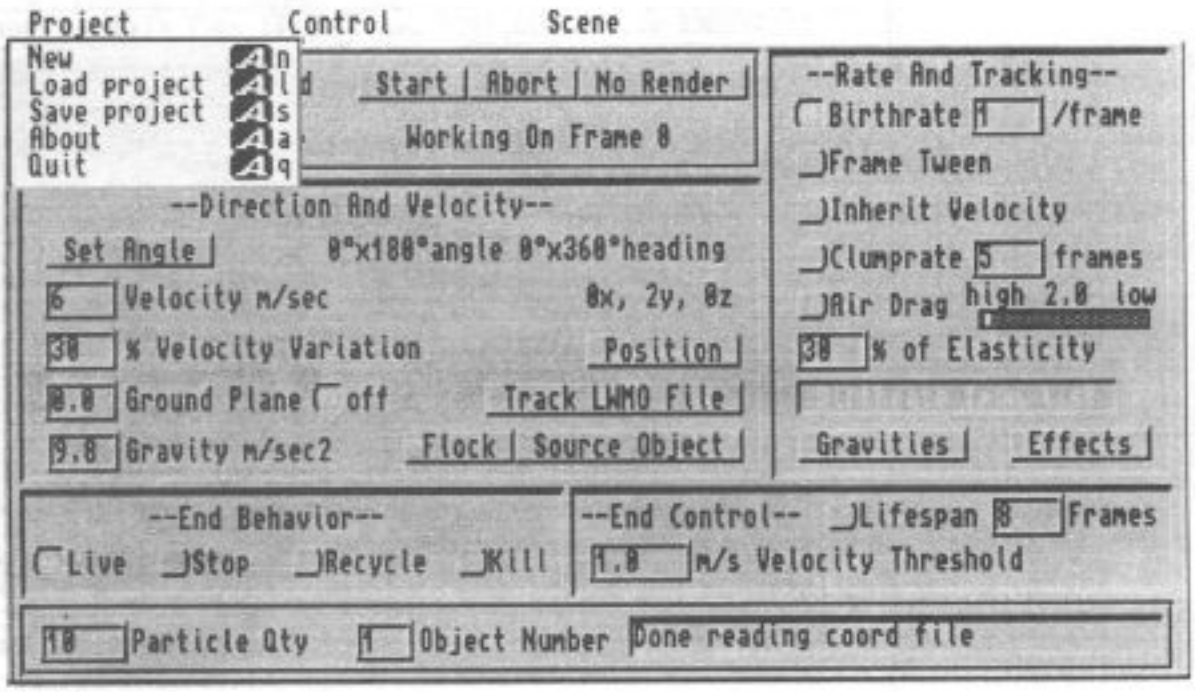
Object Number

Enter the Object number from LightWave that you wish to affect. This is the object in the layout that will have its vertices manipulated. If you are only interested in the scene file and are not manulating a object in the layout you can ignore this.

Status

Useful info about what is happening. Turns into a gauge to judge the progress of each frames calculations.

5.1 Project Menu



New

Start a new project. Resets the important buttons and clears all variables lurking around in **SPARKS** memory that may cause unplanned particle behavior.

Load project

This will load most variables into a file for the next time you want to work on a project. The important thing to remember is that although the path names to various motion files are stored they are not actually read in when a project is loaded. You will still need to click on the various highlighted buttons to setup the internal tables.

Save project

Will save most variables into a file for the next time you want to work on a project.

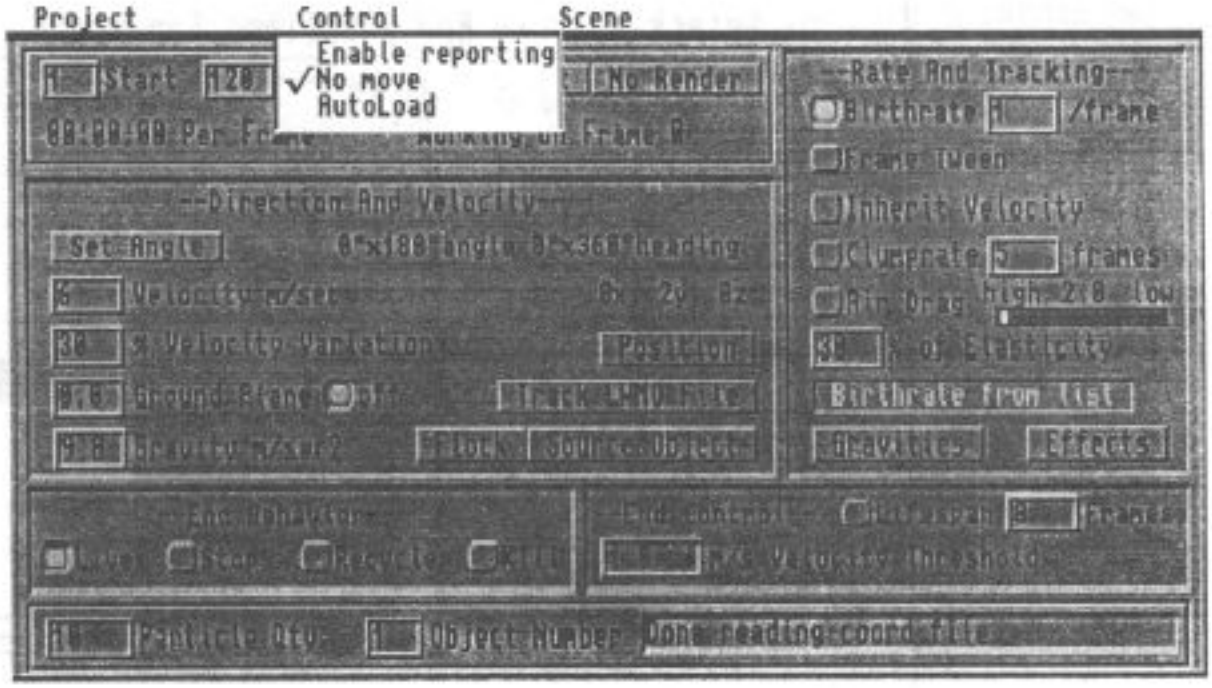
About

Serial number info. MetroGrafx tech support number.

Quit

Asks first, then quits.

5.2 Control Menu



Enable Reporting

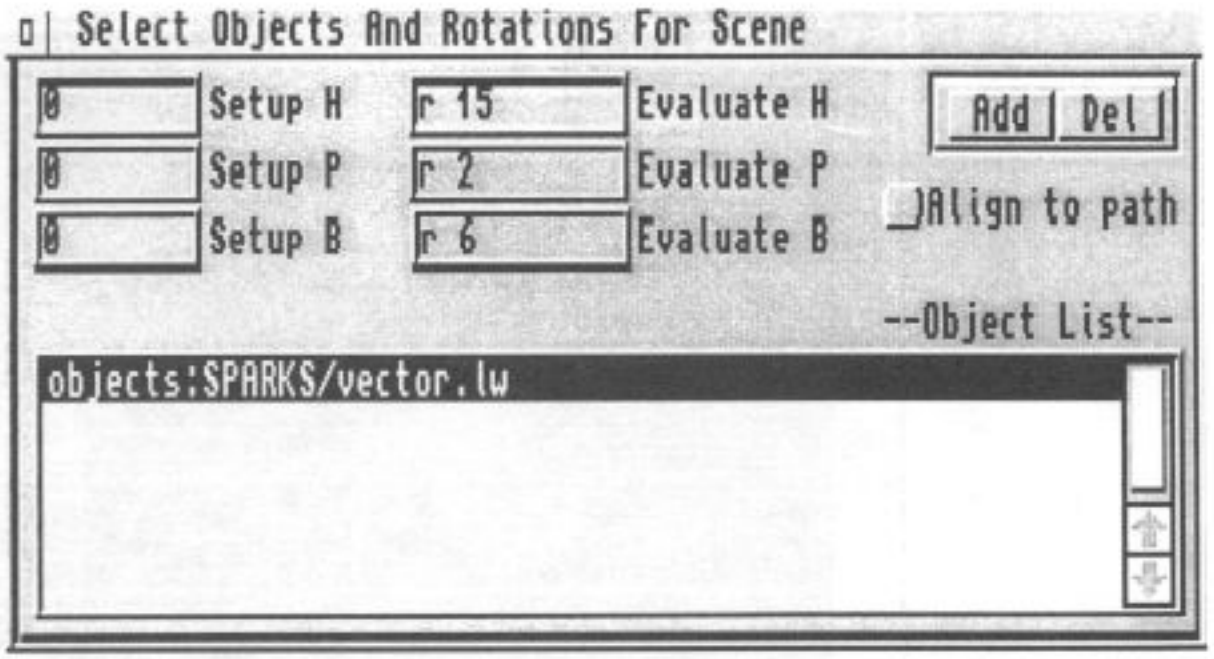
Gives more detailed is printed into the staus field on the progress of the calculations.

No Move

Prevents the ARexx call to LightWave to move a objects vertices. If you don't need to check the progress or actually render the frame this speeds things up. If this is checked and *No Render* is depressed there will be no ARexx calls to LightWave at all. This is useful when you are setting up scenes on a network.

Autoload

Check this item and when then scene is done rendering it is automatically loaded into LightWave. Anything there prior to the load will be lost.



Scene menu

Select Object will bring up a list of objects to use for the particle substitutions. By clicking *add*, a requester will come up allowing multiple model selection by shift-selecting. That is holding down the shift key while selecting the models you wish to include. In this way you could attach a different letter of the alphabet to each model with only one trip to the requester. To delete a selection just highlight the offending model and click *del*. Shift-selecting *add* itself will allow an entire directory of models to be added to the list.. Shift-selecting *del* will delete the entire list after confirming the action.

Setup and Evaluate

There is another very important aspect to this requester and that is the setup and evaluation fields. They allow rotations to be built and applied to your models as they move. This powerful feature is easy to setup once it is understood. There are two columns. One sets up the initial rotation and is labeled setup H, setup P, setup B. If a number like 90 is entered into setup H each model is rotated to 90° at the first frame. Okay, neat but so what. Each model can be rotated independently to a different angle by using a expression like i*5. i is the number of each particle. Particle 1, particle 2 ... . So what you have when that expression is evaluated is particle1*5=5 particle2*5=10, particle3*5=15...That

enables us to setup the initial rotations. The next column of fields enable rotations on each frame. That is at each frame the expression entered here is evaluated and rotation is set to the answer. The current Heading, Pitch and Bank are referenced by h[i], p[i] and b[i]. Each particle carries its own rotation info which is what the [i] is all about. So to rotate the heading 5° each frame enter h[i]+5. The powerful thing is that all manner of algebraic equations can be entered including sin() functions like h[i]+(sin(f*10)*5). This example will make a object rotate back and forth.

Here's the variables:
f = the frame number
i = the particle number(i will contain the number 1 thru point quantity)
h[i] = the heading for particle #i
p[i] = the pitch for particle #i
b[i] = the bank for particle #i

All algebraic operators may be used as well as transcendental functions like sin(), cos(), tan() etc.

Entering a "r" (no quotes), will activate automatic random rotation for that axis. They will have a default value of 10° per frame. Entering a space then a number after the "r" will override the default rotation. Example .. " r 36" will spin your object a maximum of 360° over 10 frames.

Save Scene

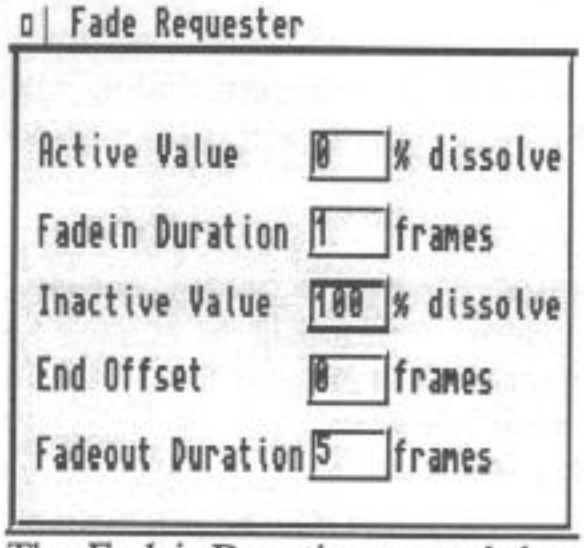
Must be checked in order to construct your scene file. When selected, a requester will load for you to select the path and name of your new scene file.

Save Motion

Simply saves particle #1's path as a motion envelope. There is no file requester as it just saves to ram:particlemotion.mot. This is handy for quickly generating a motion for a falling body, then applying it to a object, like bouncing ball. It is also suitable for loading back into SPARKS as a motion file to track. Shoot one particle up with this toggled on, load it back into *Track LWMO File* then setup 30 or 40 particles to recycle with a short *Lifespan*. This looks very much like sparkler thrown up into the air.

Setup Fade

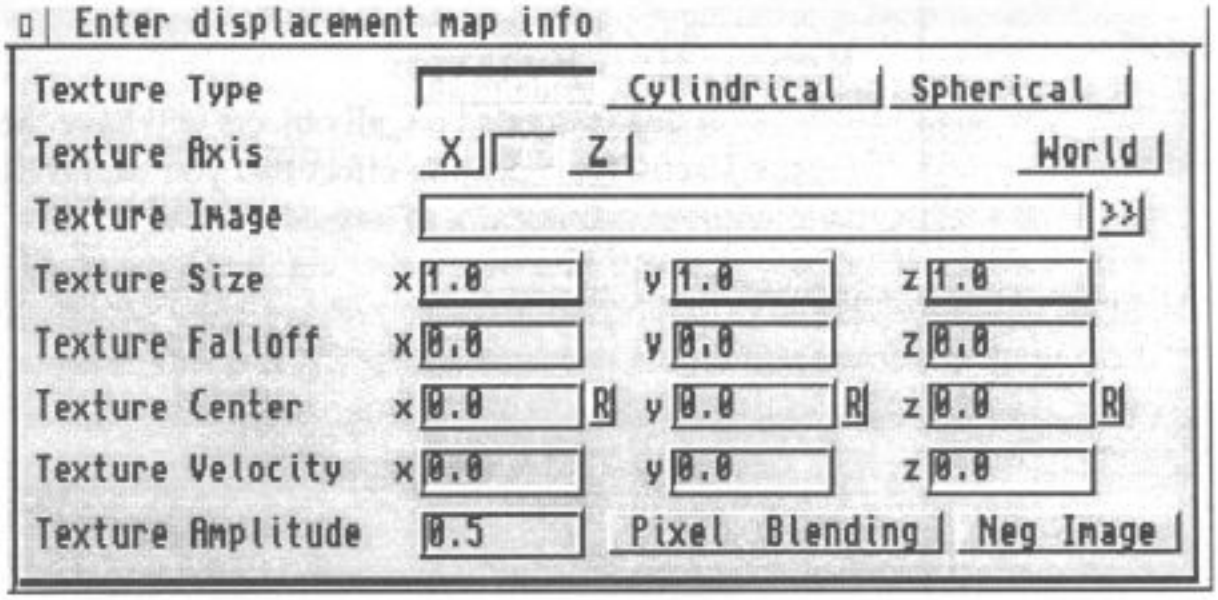
Selecting *Setup Fade* from the menu brings up the fade requester. If this menu item is checked a fade envelope will be attached to each object in the resultant scene file. This fade envelope is built on the fly for each particle depending on the particles state, active (0% dissolve) or inactive (100% dissolve).



The fade requester will enable you to custom tailor the resulting fade envelopes to your desire.
The ***Start Value*** is the dissolve percentage for the active objects.
The ***End Value*** is for the inactive state.
The ***Fadein*** **Duration** controls how many frames the fadeup from inactive takes place over.
The ***Fadeout*** **Duration** controls how many frames that the fade to end value takes place over, backtimed from the frame that the object goes inactive.
In addition the ***End Offset*** will push the fade further backward in time as it is the number of frames from the frame the object fades fully to the frame that the object goes inactive.
To turn the fade requester off so a fade envelope is not saved with your scene file just select it again. Some logic is in order here, if you are recycling the particles with a life span of 8 frames and you were to enter 3 frames *fadein* and 10 frames on the *fadeout*, that adds up to 13 frames. This would not produce the desired results. Try entering 2 *fadein* and 4 on the *fadeout* to allow for 2 frames be totally dissolved.

Setup Displacement

To make your flock of birds fly or your fish school swim, a **Displacement Map** is usually a handy thing. The problem is that they are not saved with the objects themselves but in with the scene. Since this precludes us from just loading it with the object we need to reenter the appropriate information from Lightwaves displacement requester into SPARKS displacement requester. The only different thing here is the small "R" buttons next to the *Texture Center* fields. Toggling any of



these will randomize that center value that it is next to. This makes all your fish swim, bird fly, snake slither motions offset from each other without having to go in and change it on potentially hundreds of objects. Close the panel and the menu item will be checked. If you don't want the displacement map on just reselect the menu item again.

Select Tag

This menu item will allow tagging on a bone motion file or a clip map to each object as the scene file is being constructed. The appropriate section of a presaved scene file containing the bone motion or clip to be used must be cut out and saved into a temporary file. This code is clearly labeled in Lightwaves scene file which is saved in a plain ASCII format. Use a text editor to delete everything but the applicable lines and save it. When Tag is selected a requester will come up allowing you to direct it to the tag file saved previously. Now you have Dem bones on all your models!

Parent Object

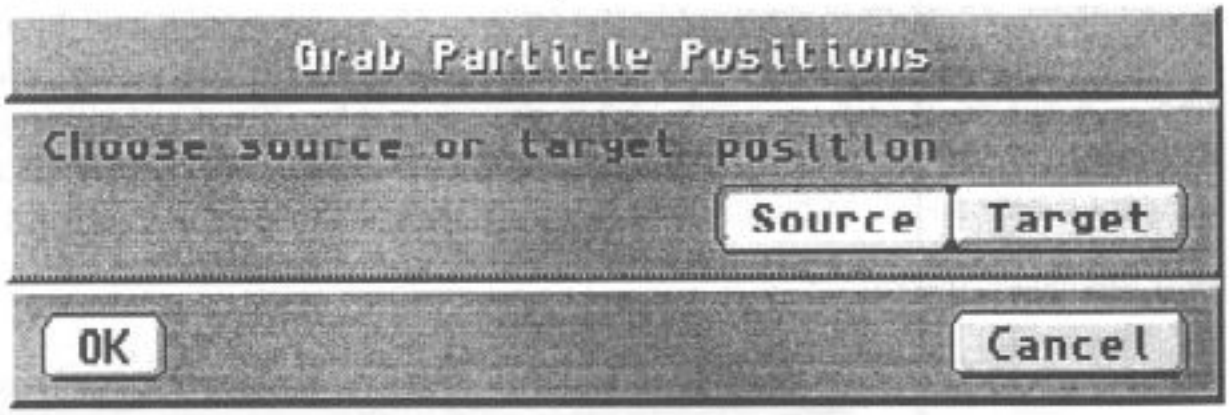
This will parent all objects to a null object. No it is easy to move all objects as a group into a new position. Deleteing the parent object will bring up a requester in LightWave asking if you would like to delete the children too.

Repeat End Behavior

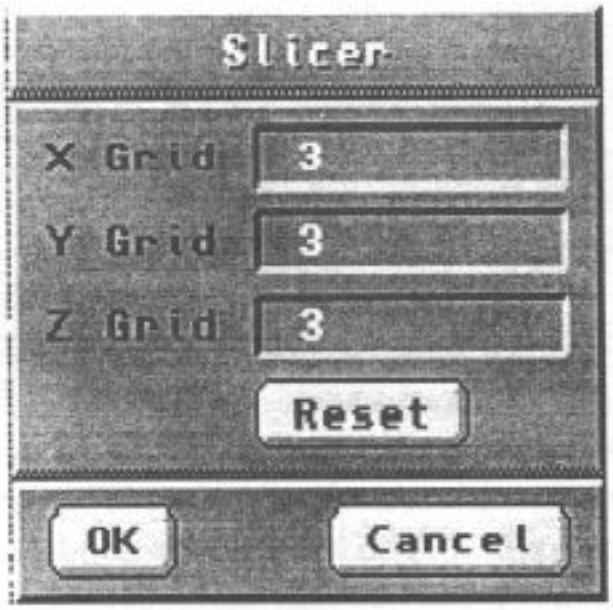
If this is toggled on, all objects will have there endstate set to repeat. Useful for repeating effect that you would like to loop over and over without making alot of keys.

Getpoints.lwm

This is the macro you will find yourself using again and again. Its purpose is to grab all the current layers points and write them to a temporary file in ram:t called either source or target depending on your choice. This is a ACSII file so if you would like to mess with the raw



numbers, load into you favorite text editor, make the changes and save. When launched you are presented with 2 choices source or target. The button to select depends if you are trying to position the initial start position (source), or the position to fire them at (target).

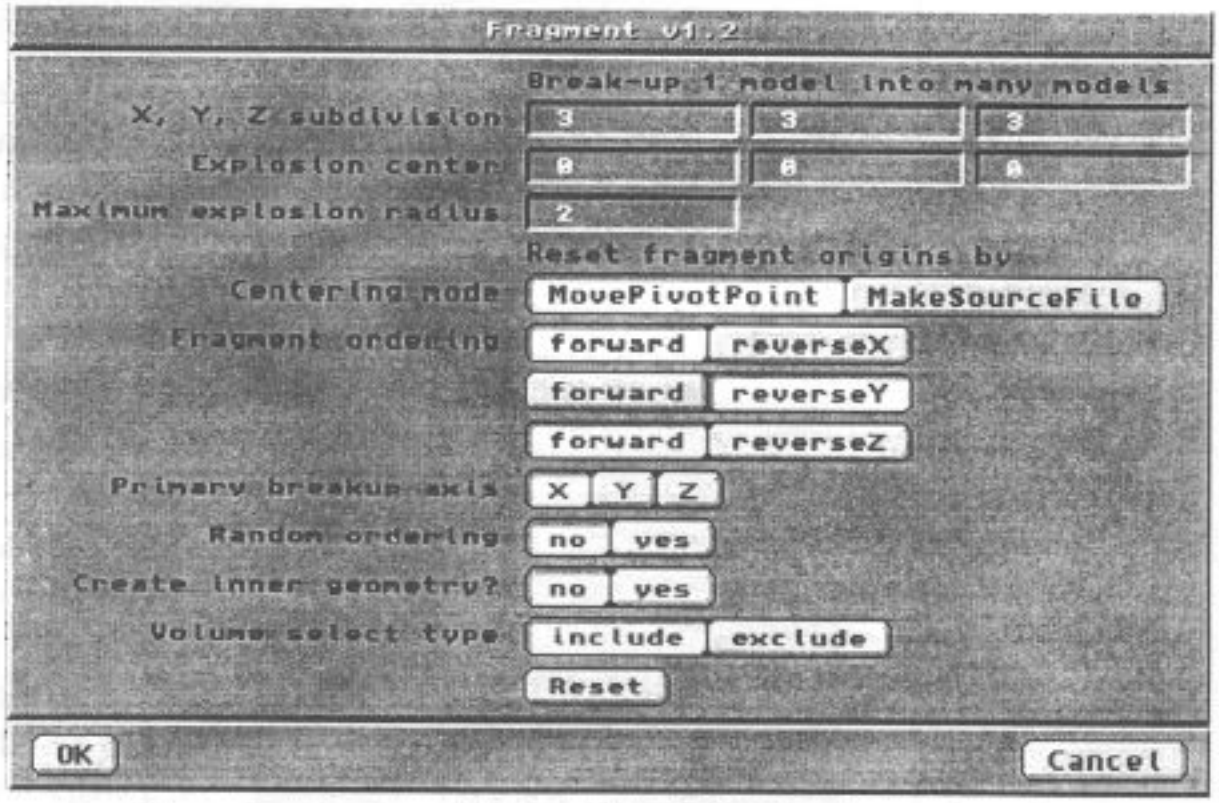


Subdivide.lwm

Slices up the current layer thus making a meshlike object with all the long polygons broken into smaller ones. Perfect for flexing with bones or breaking up the big polygons for exploding! Works by building some cutters on a spare layer then using the Modelers template drill to slice it across all axis.

Fragment.lwm

Prepare a model for exploding! Thats right, you can now explode objects with SPARKS. The gist is this. Takes the current layer with your model on it. Asks for a base filename, a subdivision amount and a explosion center. Based on the subdivision levels for x y and z, a volume select box is passed over the model selecting polygons, cutting them and pasting to a scratch layer. The center of "mass" is found and the fragment is centered based on this. This center of "mass" for each fragment is saved into a SPARKS source file.The source file repositions each fragment back into its original location for the start of the animation. The *MakeSourceFile* method has a mapping problem as follows, since the objects are all repositioned when centered you would need to have each objects surface with different offset centers in order to get the textures to line up with each other.



If you have selected *MovePivotPoint* the pivotpoint of each fragment will be repositioned with a move pivotpoint line in the scene file. Image maps are best done using this method because the objects are not repositioned in 3 space like they are with make source file.

The explosion center is used to calculate a vector passing from the explosion center thru each fragments center of "mass" that is wrote into a SPARKS target file. So that when you use the source› target option

in the source object requester the explosion force vector is directed outward through each fragments center.

If you want a model to explode where it is positioned in the layout just use the saved transformed option in LightWave to preserve the offsets and rotation and scaling. When using the *MovePivotPoint* method the value entered into the position requester moves the object about in the layout. When selecting the level of subdivision remember that xsubdivision * ysubdivision * zsubdivision = number of possible fragments. ie 3*3*3 = 27 "cuts". The number of models actually produced depends on the number of cuts that have picked up some polygons.

The fragmentation takes place down an axis normally from - to + this direction can be reversed by toggling the fragment ordering controls.

The primary breakup axis causes the fragmentation to change eroding your model down any axis. With this control and fragment ordering set correctly and *Birthrate* on, your animation can shatter, drip or shoot model fragments from top to bottom, left to right, front to back or the reverse.

If a random breakup is what you desire just toggle the *random* button.

Selecting *create inner surface* will make a inside surface and name it "inner" appropriatly enough. This way it can have a different surface.

This macro is not only great for explosions but for shatters and video type effects. Just map on moving sequences then break your panels up into poly and shoot them off.

GetSource.lwm

The title just about says it all. Puts the SPARKS source file in the current layer. This is the file that *Getpoints.lwm* saves to t:. Handy if you have deleted the source file from the Modeler and want it back to manipulate further. For instance a job I just finished called for tiles to fall from the top of the frame and end up as pieces of a panel.The panel was used to park a logo in front of. I made a rectangle the required size with segments of 20x 8y 1z. This gave me 160 polygons. I ran *Fragment.lwm* with a subdivision of 21x 9y 1z. This Split my rectangle into 160 separate models. SPARKS took these models and the source file the

fragment.lwm macro provides and setup the animation. The tiles are launched on twos over 80 frames and just push out and fall down, by turning the camera upside-down and playing the animation backward we almost achieve the look. The problem is the client wants the tiles to fall in a random fashion, not the nice neat rows that we have just done. No Prob!, just use *GetSource.lwm* to bring in the source file, cut and paste it around a bit to screw up the point ordering. Blammo! instant randomness. The client loved it. Unfortunately this technique does'nt work when you are moving the pivotpoint, as well as when the fragments are a different size. So use the random ordering in the macro to start with if you can.

GetTarget.lwm

Likewise here. Puts the **SPARKS** target file in the current layer. Real handy to use with the *Fragment.lwm* macro because it does'nt ever have its target file in a layer to manipulate. If you want to alter it you will need this just to bring it in. Then save it back out with *GetPoints.lwm*

Pathfreezer.lwm

This was a important macro prior to this release. I wrote this to take a motion path saved from LightWave and freeze with a key at every frame coinciding exactly with its free counterpart, taking into account tension, continuity, bias. Your raw motion file haphazardly keyed will emerge renewed with keys suitable for **SPARKS** delicate inner workings. Enter the original motion file and the macro prompts for a new name. Just feed the transformed file into *Track LWMO File* or when *Flock* prompts for it and you are in business. **SPARKS** now can do the job of this macro itself but it is included here because some still may find it useful.

7 Tips and Tricks

Things to try? tips & tricks? hints?

Preview you particle movement with small number of particles to check if they are doing what you want.

If you are using a source object to setup the initial particle origins, be sure they are above the ground plane value. If they are "below ground" they will be moved to 0 when they become active.

If you are intending to recycle your particles and writing out a scene file so you can render with motion blur, you should also make a fade envelope. When the particles go inactive and move to the current origin in order to be recycled they will leave a particle trail. This trail goes zipping back up to the origin in 1 frame and leaves a objectionable flickering artifact. If the trail is short and the origins in a container it can sometimes be masked. If there is a fade envelope however it can be completely eliminated.

The objects used for replacement can be used for many effects. These objects don't just have to be normal models. Try using groups of points as models or lines (2 point polygons). If you use a displacement map the internal points on a point mass can move even as you are moving them, ala fish or birds to smoke and fire.

For a cool fire effect for example say from a log, make a log mesh in the modeler. Save the model. Kill the polygons. Cut away the point where you do not want flames to originate. Run the getpoints.lwm macro. Make several trianglular flamelet objects of appropriate size and give them bright orange, yellow and red colors. Set luminous high. In **SPARKS** use the source object, set the angle upward, gravity 0, set recycle on, recycle on frame 8-12 or so. Set the birthrate to a number appropriate to the number of points you are trying to cover and the number of flamelets you have set. The camera should be set for motion blur on 100%. This will give the flames transparency as they shoot upward. Setup a few lights of warm colors and jack their intensity around to get some flickering. For a real treat, displacment mapping the flamelets as they go up will make them dance around for you.

Colors can be altered by mapping. If the map is a world map the particles will move through it changing color as the map does. The map can also be a color gradient planar mapped to particle that has a velocity that will change the particle color as it moves.

Scenes can be appended together using the load objects from scene button in Lightwaves object menu.

Objects can be prevented from bouncing by setting the bounce probrobility to 0 or by setting the velocity threshold to a number much higher then the value you would anticipate the collision to occur.

Particles can cast shadows using spotlights and shadow mapping.

Normally if the particles are not moving they are prevented from rotating with the evaluation function. There is a special case though and it occurs when the velocity is 0 and the gravity is 0. In this special case rotation evaluations are enabled. This allows great effects with arrays of objects. Set up a grid of objects and they can all hang in space rotating in or out of sync depending on their initial start rotations. Pan the camera across with a dramatic lighting effect, slide in a logo and instant broadcast look.

Particles always come to rest on the ground plane if the *Lifespan* toggle is off and the *Ground Plane* is on. If your objects are penatrating the ground object in your scene, either move the ground object down or set the *Ground Plane* value to up to compensate and recalculate the scene in SPARKS.

Local gravity wells can be animated on a path. Using a anti-gravity they can push around objects laying on the ground plane with a *live end behavior*. This looks great, kind of like pushing sand around on a table with your finger tip. If the gravity is traveling fast and underneath it can slam into them pushing them up and over like a wake of a boat. I made a animation placing particles on the gound plane. I made a motion file representing the gravities path slightly under the particles position. Then I loaded the path on to a anti-gravity well. Saving the scene from SPARKS with leaves replacing the particles and swirling on I added in a street object and a motorcycle. Running the

motorcycle across the same path as the gravity was resulted in a incredibly real animation of leaves kicked up by a motorcycle passing by.

Smoke is possible with **SPARKS** but as with any advanced effect it is really an integration of several techniques. Split the layer to contain the smoke onto its own animation layer. Render it out with no anti-aliasing. Blur the sequence heavily, real heavily, like 10 or 15 times in ADPRO. Maybe use the displace pixel operator to fatten up the haze. Then, raise the gamma as required to build up the density. Use this as a matt sequence to composite white or grey over the rendering or live action. The particles should move slowly and fade as they reach the edge of the effect. Try remapping the colors on the matt sequense. By remapping the whites to yellow and the darks values to red, a fire-like glow is achieved.

Simply scale all frames to slow down a **SPARKS** scene.